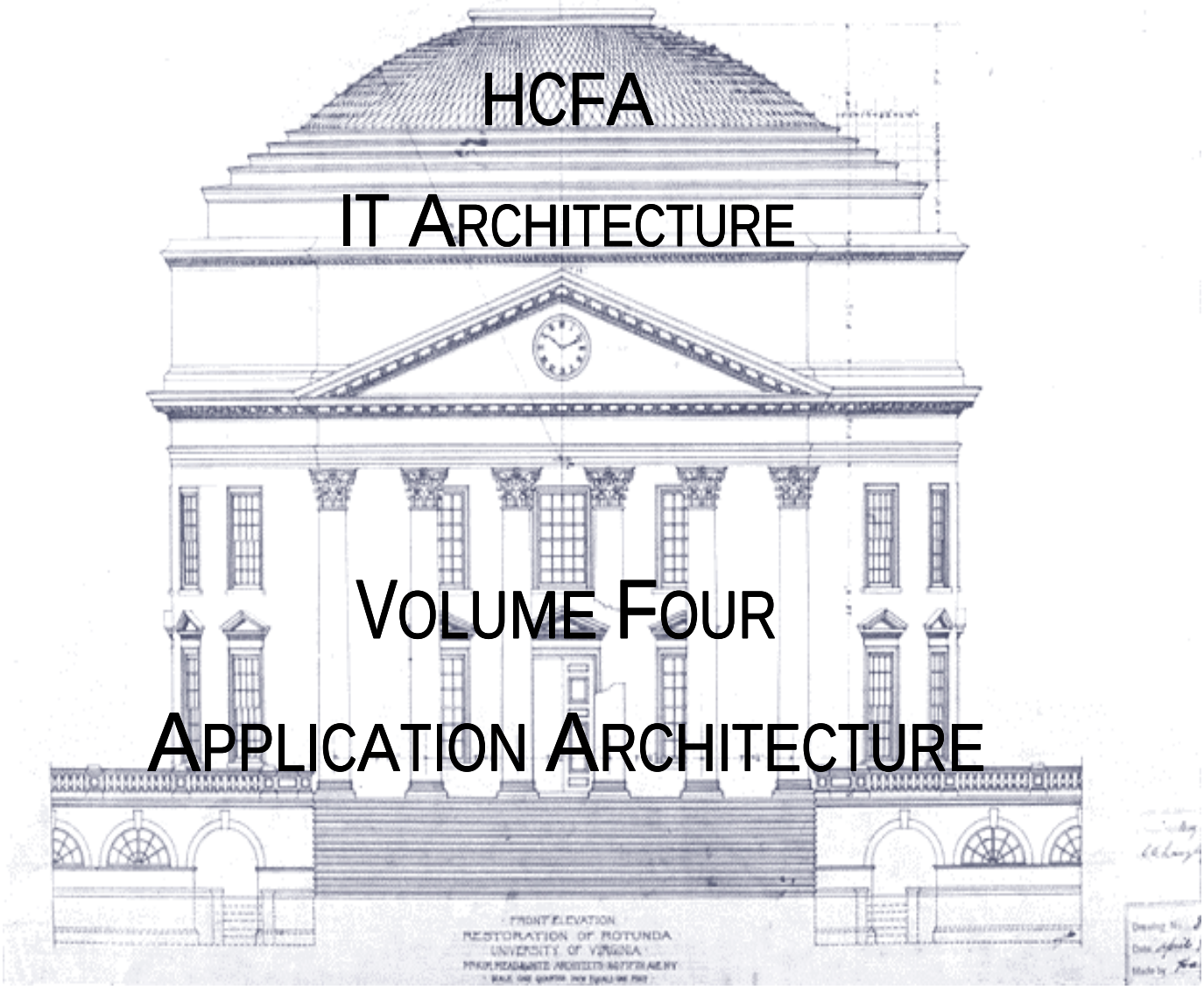
HCFA

IT ARCHITECTURE



VOLUME FOUR

APPLICATION ARCHITECTURE

Version 2.0

November 1999

HCFA IT ARCHITECTURE

Health Care Financing Administration

CONTENTS

LIST OF EXHIBITS

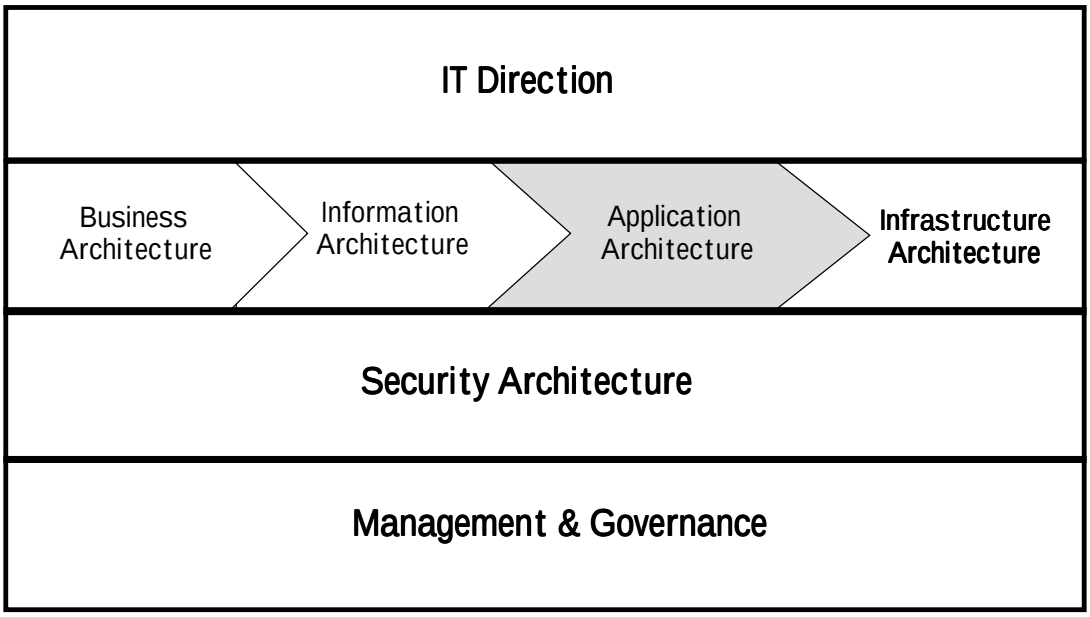
Volume 4 Application Architecture

4.1 Introduction

This volume describes the Application Architecture component of the HCFA Information Technology Architecture (ITA). The purpose of the Application Architecture is to provide guidance in creating or acquiring business applications that are consistent with the IT strategic vision and direction as described in Volume 1 – IT Direction.

This volume should be used in conjunction with Volume 2 – Business Architecture, Volume 3 – Information Architecture, and Volume 5 – Infrastructure Architecture, to ensure full understanding of the references made throughout this volume, and to place the interrelationships with the other architectures in the proper context. Exhibit 4-1 depicts the relationship of the Application Architecture to the overall HCFA ITA.

EXHIBIT 4-1. INFORMATION TECHNOLOGY ARCHITECTURE COMPONENTS



Questions concerning the contents of this volume should be directed to the Office of Information Services, Information Technology Architecture Staff. An online version of this and all other ITA volumes can be found on the HCFA ITA Intranet and Web site at http://hcfanet.hcfa.gov/hpages/ois/ita.

4.1.1 Background

HCFA is responsible for a variety of business functions and processes that enable the Agency to administer health insurance to over 70 million Americans. Its programs and policies are of national importance to the health care industry. The highest-volume business process carried out by HCFA involves the processing and payment of claims filed by providers on behalf of beneficiaries. The prompt and accurate processing and payment of an increasing volume of claims, and demands for more data and information access, pose a major IT challenge for HCFA. The advance of managed care and the structural changes that are moving the health care industry away from traditional fee-for-service plans also pose new IT challenges. Another significant function performed by HCFA is maintaining and providing health care information for service delivery and policy decision-making. In addition to supporting the information needs of internal HCFA staff, the Agency must also respond to the needs of HCFA customers and other entities that require and have a right to information on Agency processes, procedures, and data. These include oversight bodies like Congress, the Office of Management and Budget, the General Accounting Office, and the Office of Inspector General, and external data customers such as universities, researchers, and Freedom of Information Act (FOIA) requesters.

Software applications are the primary IT tools used to access the information needed to support Agency business. In the 30 years since the inception of Medicare and Medicaid, information systems technology has undergone extraordinary change. Today, significant gaps exist between HCFA's current business needs and the performance of its current business applications. (See Volume 1, Attachment A – HCFA's IT Vision, for a discussion of these gaps.) Consequently, HCFA's business applications must evolve to strategically fill these gaps. The Agency's business applications must be able to handle current business needs; easily expand to address future needs; readily support the administration of new programs; and seamlessly adopt new, more efficient technologies. In essence, future business applications developed for HCFA must enable business change in a timely manner.

4.1.2 Purpose and Scope

The purpose of the Application Architecture is to provide guidance in creating or acquiring business applications that are consistent with the IT strategic vision and direction as described in Volume 1. The Application Architecture is not intended as a detailed design specification, nor is it a systematic procedure for designing applications to support specific business needs. Rather, it provides a common conceptual framework that IT investment decision-makers, project planners, systems designers, and application programmers can use to:

- Identify the logical information systems needed to support HCFA's business processes, and relate those systems to the databases to which they provide access;
- Make more informed decisions (of a business or technical nature) about when to invest in new information systems development efforts;
- Leverage IT investments by designing future applications that are adaptable, more maintainable, and reusable;

- Encourage standardization of the technical infrastructure needed to support HCFA's business applications; and
- Establish an assessment methodology with which to evaluate and determine the disposition of HCFA's physical legacy applications within the target architecture.

The scope of the Application Architecture extends to all business applications within HCFA that are mission-critical or of enterprise importance. However, this guidance need not be limited in this manner. Any IT projects involving the design and implementation of business applications within HCFA can also benefit from the concepts and approaches set forth.

The remainder of this volume is organized as follows:

- Section 4.2 describes HCFA's strategic direction by contrasting the disadvantages of the current legacy applications with the benefits of the target Application Architecture.
- Section 4.3 presents HCFA's approach for migrating/evolving from the current legacy environment to the target Application Architecture.
- Section 4.4 presents the design principles and recommended best practices to be followed in designing and developing applications in the target environment.
- Section 4.5 discusses considerations for HCFA regarding the development/acquisition of specialized skills to enable the successful delivery of IT services in the target environment, as well as some organizational considerations.
- Section 4.6 describes the standard methodologies and tools to be used in developing business applications in a manner consistent with the design principles established for the target Application Architecture.
- Section 4.7 presents policy guidance to ensure that HCFA achieves the objectives embodied in the target Application Architecture.
- Section 4.8 discusses future activities related to assessing HCFA's legacy application portfolio with an eye toward evolving to the target Application Architecture.
- Section 4.9 contains a feedback form with which readers of this volume can provide comments and feedback pertaining to the target Application Architecture.

4.2 Strategic Direction

Information — specifically, the collection, distribution, and analysis of information — is the key to HCFA successfully meeting its strategic objectives. Agency activities that are dependent on accurate information include certifying the eligibility of beneficiaries, correcting claims payments, determining policy, and assessing health care outcomes. HCFA's ability to successfully accomplish its mission is dependent on reliable business applications, data, and supporting infrastructure.

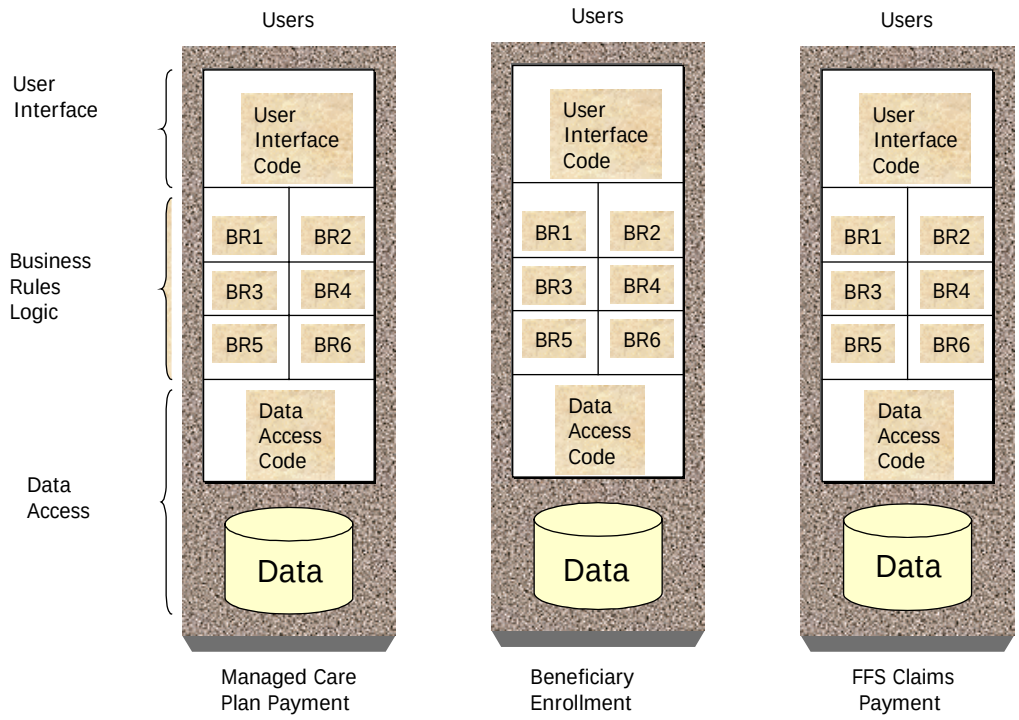
HCFA's IT Vision[1] describes an information-centered approach in which applications work more effectively by sharing information. This approach will ensure that HCFA is more responsive to changing business requirements and emerging technology. To accomplish this vision, HCFA will establish enterprise databases in which applications utilize standard interfaces for accessing data and information. Future applications will be designed with well-defined boundaries and will be more modular, portable, reusable, and easier to maintain. The HCFA Application Architecture provides a strategic approach for the implementation of this vision.

4.2.1 Legacy Application Environment

HCFA's current application and IT environment is composed primarily of legacy systems, which is typical of many Federal civilian organizations. The HCFA environment consists of over 100 such systems. Legacy systems tend to be monolithic in their design and construction. This is because they were designed from a departmental or divisional view and not from an enterprise perspective. The end result was the creation of many stovepipe application systems. HCFA's Group Health Plan (GHP) system, Enrollment Database update system, and standard claims processing systems are examples of this legacy design. Exhibit 4-2 is a simplified illustration of typical monolithic stovepipe systems.

[1] HCFA Information Technology Vision, July 30, 1998, by Gary Christoph, Ph.D., Chief Information Officer

EXHIBIT 4-2. STOVEPIPE APPLICATIONS



4.2.1.1 Monolithic Applications

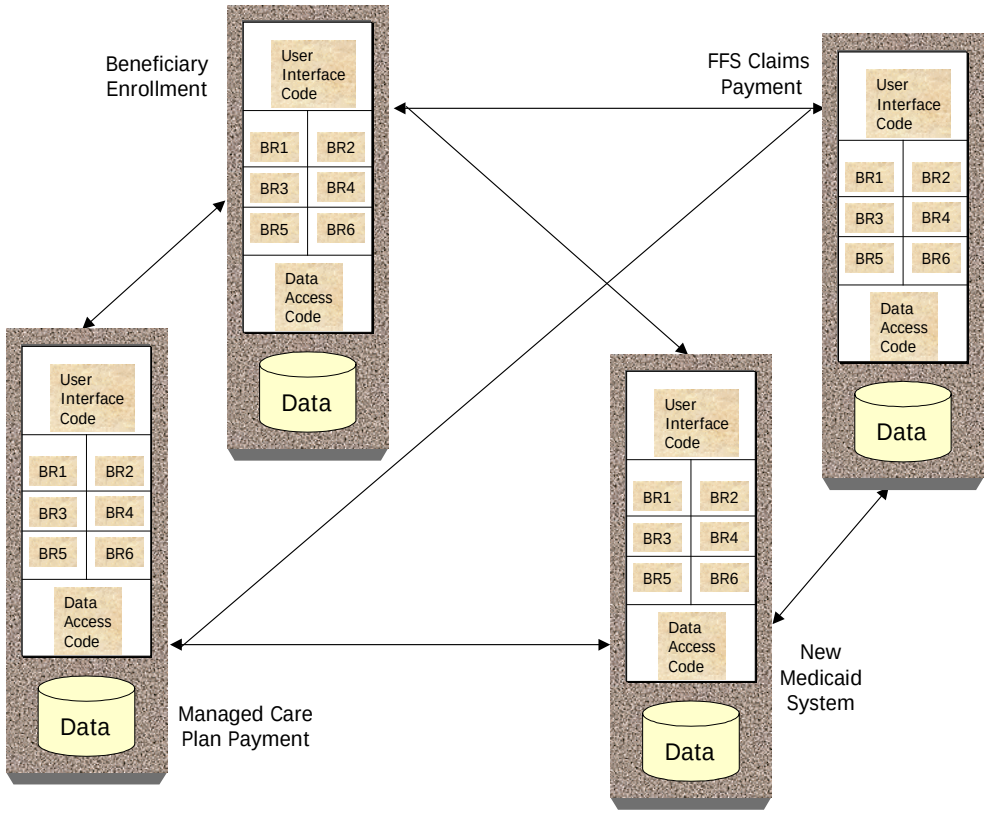
HCFA's applications support multiple business processes using complex, tightly coupled business rule logic and customized data access methods. This approach introduces a number of problems. These include:

- **Applications That Are Too Complex and Costly to Modify and Maintain** – Because of their size and tight coupling, and the tendency to develop into spaghetti code, monolithic application systems are inherently costly and difficult to modify and maintain. Even small changes take a longer time to implement and test, resulting in higher costs.
- **Limited Reuse of Application Programs and Processes** – Because of a lack of modularity, each new system requires the recoding of business rules. This increases the probability of introducing errors, increases coding and testing time, and requires new documentation, all of which drives up costs. Because each monolithic application system is managed independently by different HCFA business units, common functions are implemented differently. This makes it difficult for HCFA systems to reconcile information, and increases enterprise-wide software maintenance costs.
- **Difficulties Integrating and Sharing Data and Information** – Although much of the data accessed by one of these systems may be identical to data used by other

systems, the databases are not shared. The multiple, disparate databases containing beneficiary-related information that currently exist throughout the Medicare operating environment are indicative of this problem.

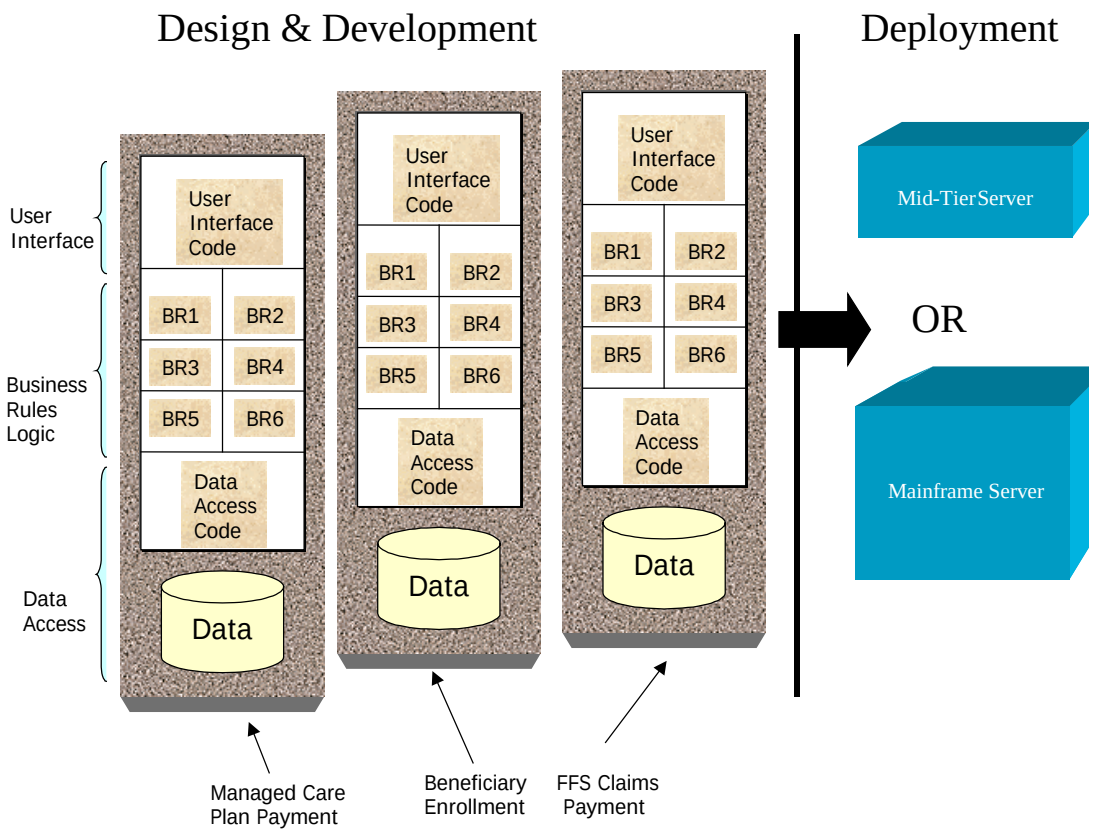
- **Proliferation of Disparate Batch Files** – Batch flat files of dissimilar formats are the primary means of sharing data and information between one business unit system and another. Adding a new system to support a new business process usually necessitates creating custom batch files as interfaces between systems. This continued practice increases resource demands on the infrastructure and adds to the complexity of data management throughout HCFA. Exhibit 4-3 simplistically illustrates how batch file sharing can easily proliferate into a spider web of flat file interfaces.

EXHIBIT 4-3. PROLIFERATION OF BATCH FILE INTERFACES



- **Rampant Data Redundancy and Inconsistency** – Proliferation of disparate batch files increases HCFA's data management burden. Over time, if more batch processing systems are added and HCFA data volumes continue to increase, data timeliness and quality is further degraded, exacerbating the situation.
- **Inflexible Deployment Alternatives** – The practice of making decisions regarding the hardware platform on which to deploy an application before it is designed and developed imposes artificial constraints on systems design. This practice limits deployment alternatives once the application system has been developed. Typically, monolithic applications employ a tightly coupled design that binds the user interface logic, business logic, and data access logic into a singular system. Consequently, the system must be deployed onto a given hardware platform (either the mainframe or a mid-tier server), and cannot take advantage of other available infrastructure resources within a distributed environment (see Exhibit 4-4).

EXHIBIT 4-4. STOVEPIPE SYSTEMS DEPLOYMENT



4.2.1.2 Difficulty Leveraging Information Assets

Many of HCFA's IT assets (i.e., applications, data, and infrastructure) are created for use by one business unit and are not available to other business units, resulting in little sharing and reuse of these assets. IT requirements to support new business needs tend to involve creating "new" applications and "new" data, even though similar application processes and data may already exist elsewhere. This causes two main areas of difficulty:

- **Limited Awareness of Existing Application Processes** – Many of HCFA's current business functions carry out similar processes and therefore need access to similar data. However, when application systems are designed to support the business processes of one business unit, the design usually does not take into account other uses within HCFA, leading to the creation of dissimilar applications. It is difficult for systems analysts and designers, during the design phase of their projects, to identify existing application processes that provide similar functions and data access, and so it is difficult to avoid designing redundant solutions.
- **Limited Awareness of Existing Data and Methods for Access** – Like the applications, many of HCFA's current business functions are similar and need access to similar data, yet during the design of the application systems, data created for use by one business unit usually is not considered for other uses within HCFA, leading to the creation of redundant and often conflicting data stores. It is difficult for data administrators and database designers, during the design phase of their projects, to identify existing data stores and access methods that are similar to their needs, and so it is difficult to avoid creating more redundant data.

4.2.1.3 Difficulty Leveraging Roles and Responsibilities

Not all problems with the legacy application environment can be attributed solely to the monolithic design of systems; another factor is the manner in which HCFA apportions responsibility for application systems delivery. Our current application delivery model can be characterized as a stovepipe approach, where the responsibility for IT project delivery is held solely within a business unit rather than shared at the enterprise level. Business units assume full responsibility for all aspects of a system implementation, from conception through design and deployment, including some operations support, which typically falls to project leaders, systems analysts, systems designers, and programmers. One person often assumes multiple (and in some cases, all) roles and may lack the requisite time, experience, or resources to carry them out effectively. Some examples are:

- **Project Leader** – Project leaders often carry broad responsibility, including project management, contract management, and system development life cycle (SDLC) activities within a business unit. Because of this insulation, the sound project management practices, contracting approaches, or software development techniques created by one business unit are not easily leveraged by other business units. The

relative immaturity of key SDLC processes[2] throughout HCFA places a tremendous burden on IT project leaders executing projects within a business unit.

- **System Designers** – Systems analysts within HCFA business units are generally knowledgeable of specific legacy applications and their operations. The user interface, data access methods, languages, hardware platform, and operating systems were predetermined to be mainframe oriented, for the most part. Continuing with monolithic system designs may prove expedient in this environment, but such a practice is problematic for designers not familiar with integrating multiple technologies. HCFA's experience with designing applications to operate in a heterogeneous, distributed environment is isolated and targeted to specific business unit needs. An innovative system design created by one business unit is not easily leveraged by system designers outside the business unit.
- **Programmers** – Programmers developing HCFA systems usually are required to perform or take part in every process within the SDLC, regardless of the fact that many of the key processes require competencies that are different from pure application programming. Programmers are assumed to be experts in all aspects of systems design and implementation, including user interfaces, business rule logic, network interfaces, and data management operations. Innovative programming techniques or common program routines created by one business unit are not easily leveraged by other business units. A tendency among programmers is to tightly couple data access logic with business logic and user interface logic within an application. Consequently, simple changes in data can negatively impact other parts of an application, causing a ripple effect throughout the system. Data management problems and the responsibilities of database administrators are compounded as a result of this approach. As advances in technology also increase complexity, programmers can no longer be expected to maintain proficiency in all aspects of application development.

4.2.1.4 Inhibited Ability to Change

For the most part, HCFA business applications remain substantially similar to technology implemented 10 to 15 years ago, despite operating on newer equipment. Over time, the legacy application environment has produced many inhibitors that adversely affect HCFA's ability to respond to demands of a changing Medicare and Medicaid business environment. HCFA's legacy environment, encompassing hundreds of business applications n created over a period of 30 years, cannot be transformed overnight. Our rich and valuable data and information resources have become tightly bound in the legacy systems infrastructure. Our ability to quickly adapt IT to changes in business and information requirements is inhibited by our inability to change the legacy application systems, which consequently inhibits our ability to change the infrastructure. The negative impacts on IT effectiveness, and the frustration of HCFA data users as a result of these inhibitors, have steadily increased over time. Just as it took decades to get

[2] Key processes and process maturity as defined by the Carnegie-Mellon University Software Engineering Institute (SEI) Capability Maturity Model (CMM). More information can be located on the SEI/CMM Web site at: WWW.SEI.SMU.EDU. Note: HCFA has not formally been assessed under the SEI CMM method.

to the current state, it will take years for the legacy application environment to evolve and be transformed to make the most effective use of information resources. A new paradigm for designing and implementing business applications is needed to avoid perpetuating this legacy.

4.2.2 The Target Application Architecture

This section describes the strategic vision and primary characteristics for future applications developed to support the changing business needs of HCFA.

4.2.2.1 IT Vision Alignment

HCFA's IT Vision describes an environment in which existing and new application systems work more effectively by sharing information, and in which HCFA is more responsive to the demands of changing business needs and the promises of emerging technology. It represents a shift from a process-centric paradigm to one where information provides the orientation for the technology infrastructure. HCFA's IT Vision can be characterized as an information-centric model that emphasizes three main elements:

(1) Data management is a core function and data is treated as an enterprise asset.

(2) Individual business functions are supported by modular applications that are reusable across program areas (applications are designed to perform discrete operations against data).

(3) All databases are readily available to the business functions through standardized interfaces.

While the Information Architecture is aimed at fulfilling element 1 above, the target Application Architecture is aimed at fulfilling elements 2 and 3 by providing a framework for the evolution of current and future business applications at HCFA in a manner consistent with the Agency's IT Vision and strategic direction.

By defining a set of IT Objectives and IT Guiding Principles, the ITA sets forth the direction in which HCFA will proceed toward achieving the IT Vision. Eight IT Objectives describe the desired future-state IT environment for HCFA. Fifteen IT Guiding Principles provide broad guidance for how we will go about achieving the desired future state. (See Volume 1 for a detailed description of the IT Objectives and IT Guiding Principles.) Collectively, the IT Vision, IT Objectives, and IT Guiding Principles provide the philosophical underpinnings for all of the ITA component architectures, including the target Application Architecture.

4.2.2.2 Characteristics

Future applications must be adaptable to the changing business needs of HCFA in a timely manner. Therefore, *adaptability* is a primary design characteristic for applications in the target environment. *Adaptable* applications are characterized by the extent to which the following features are exemplified in the design and operation of systems:

- **Flexibility** – Applications in the target environment must be easily extensible to incorporate new business requirements and to take advantage of technology innovation with minimal effort. Flexibility leverages investments in applications by shifting resources from maintenance activities to development. Over time, application flexibility reduces the level of effort necessary to implement new requirements in response to changing business needs.
- **Maintainability** – Applications in the target environment must be designed and developed for ease of maintenance. Simplicity is preferable to unwarranted complexity. Limiting the scope of a single application to a discrete function performed on a single data entity allows for modularity in design, which reduces complexity.
- **Reusability** – Applications in the target environment must be developed with reuse as a primary design point. Applications constructed of modular design components are conducive to reuse. Reusable components can be shared by other applications within a business unit or by other business units throughout HCFA. Reusable components, defined as common and shared application and infrastructure services, can be made available to other applications throughout the enterprise.
- **Portability** – Applications in the target environment must be portable to different operating platforms, if necessary, with minimal effort and without redesign. Portability enables applications to be deployed wherever they may be needed by HCFA users to access enterprise information or to optimize the use of available computer resources.
- **Scalability** – Applications in the target environment must be scalable to accommodate increased numbers of concurrent users accessing data, as well as increased volumes of transactions and increased database size. Scalable applications can adapt to dynamics within HCFA's business environment without adversely affecting user productivity or system operations.
- **Interoperability** – Applications in the target environment must be capable of accessing databases and services across infrastructure platforms. Adaptable applications can be placed entirely onto one platform, or their components and services distributed across platforms, depending upon business and operational considerations. Applications should not have to be located on the same physical platform as a database in order to access it.
- **Manageability** – Applications in the target environment must be capable of being controlled using automated technologies for managing distributed computing environments. Managing distributed computing includes services for software version control and distribution, installation, invocation, security, monitoring, statistics, alarms, and shutdown. Adaptable applications allow these services to be implemented in a standard way so that changes to a business function do not adversely affect operations.

Creating future applications and modifying legacy systems to embody adaptable characteristics requires an Application Architecture approach that redefines how HCFA IT staff have traditionally viewed application systems. Our current applications must be further analyzed and

assessed to determine the data operations they perform, how groups of legacy systems relate to one another, and the viability of redesigning the legacy systems for consistency with the target environment.

Applications serve a variety of purposes in supporting the business needs of HCFA. Different strategies, design criteria, and technologies are available to implement IT solutions, depending upon the distinct type of application needed. For example, designing and developing a transaction processing application and database differs in approach from designing and developing a decision support application or an office automation application. Decision support and office automation solutions can be implemented using commercial-off-the-shelf (COTS) products, for the most part, with minimal customized software development. HCFA transaction processing applications, on the other hand, often require custom features and functions in order to support Agency business processes, and cannot be implemented using COTS products. The target Application Architecture offers guidance for efforts aimed at designing and developing transaction processing business applications. Application business functions are discussed in more detail in Section 4.3.2.

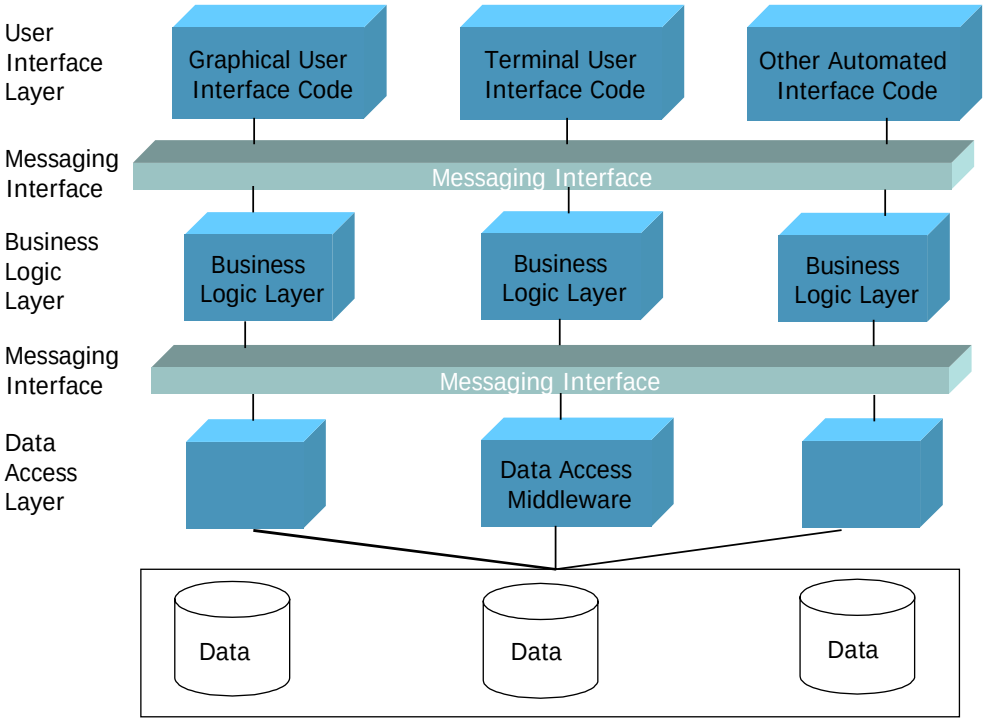
The creation of monolithic stovepipe applications in the target environment must be avoided by establishing reasonable guidance for determining application design boundaries. Two basic premises are advanced for this purpose: (1) an application should mimic the business process it supports, and (2) an application should perform a discrete operation against data. These basic concepts have broad implications for our approach to designing future business applications in the target environment. They are discussed in more detail in Sections 4.3.2 and 4.3.3.

The conceptual approach to designing future applications that are consistent with the target Application Architecture is discussed in Section 4.3. The Application Architecture provides a conceptual description of how HCFA business applications should be designed and how they will cooperate with one another. This is accomplished through the promotion of a conceptual framework that information systems project planners, systems designers, and application programmers will use to design, develop, and integrate HCFA applications. The Application Architecture framework defines standard components and interfaces that are consistent with HCFA's vision for the target architecture.

4.3 Approach

The Application Architecture is not intended as a detailed design specification, nor is it a systematic procedure for designing applications to support specific business needs. Rather, the Application Architecture provides conceptual views that represent how the application services fit together and collectively provide greater value than the sum of their individual parts. The target Application Architecture is depicted in Exhibit 4-5 below. The services of this architecture are discussed in Section 4.3.4.

EXHIBIT 4-5. TARGET APPLICATION ARCHITECTURE



Our approach in this volume is to view the Application Architecture through several different conceptual views. These views are intended to guide how future applications will be:

- Designed to correlate to discrete business processes (see Section 4.3.2, Information Systems Groups, and Section 4.3.3, Application Boundaries);
- Developed and implemented as smaller, more modular applications that are easier to modify and can be adapted to changing business requirements (see Section 4.3.4, Application Architecture Services Model, and Section 4.3.5, Application Design Concepts); and
- Categorized by their salient characteristics (see Section 4.3.6, Information Systems Design Models).

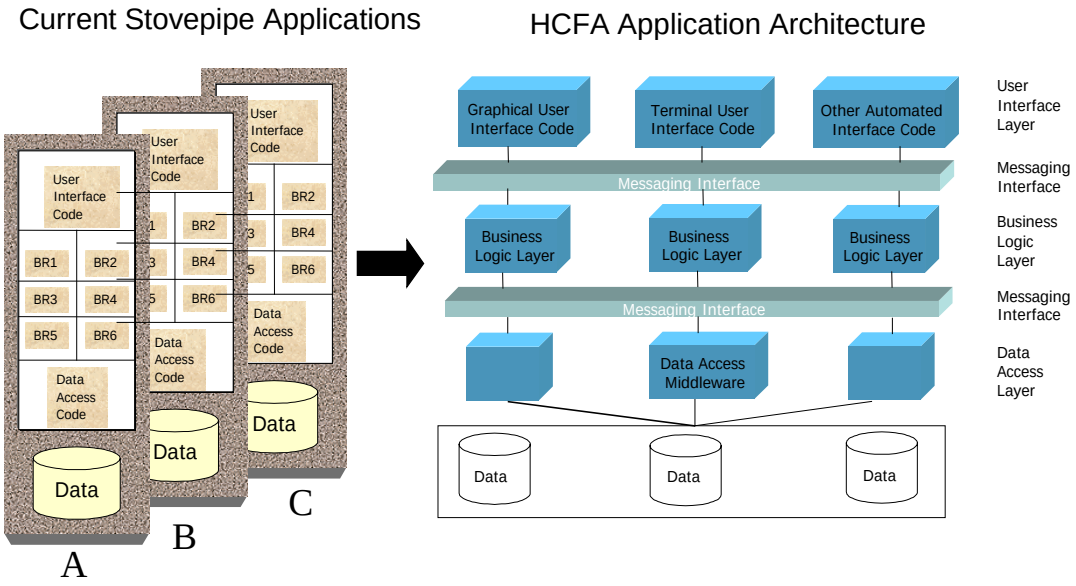
In general, applications serve two primary purposes: (1) they perform routine business functions that support a business process, and (2) they access, process, and/or display data. At the highest level of abstraction, then, applications can be organized by the functions they perform and the data they process. This organization of applications by functional operations and data creates a business view of the Application Architecture.

An application environment is a complex synergy of cooperative, integrated parts working as a whole to support the business of the organization. For an application to run, it requires access to platform resources such as a CPU, I/O devices, memory, user interfaces, security, and many others. Therefore, a framework is necessary that delineates boundaries and describes interfaces so that applications can access and use those component services in the most optimal manner. This framework for the organization and structure of component services presents a technical view of the Application Architecture.

To isolate and contain the impact of change on business applications in the target environment, an approach based on loosely coupled, layered design is needed. Loose coupling involves separating an application into modular components and enabling the components to operate as a whole using messaging techniques. Application components can be modified more easily without affecting other components, thereby reducing the level of effort needed to test and integrate changes. While this design approach is used extensively in the industry today and is not cutting edge, it represents a dramatic change from the tightly coupled, monolithic design of most HCFA legacy systems. Exhibit 4-6 illustrates a conceptual view of the transformation from a monolithic, stovepipe application to an n-tier application.

EXHIBIT 4-6. EVOLVING FROM STOVEPIPES TO N-TIER APPLICATIONS

The maximum benefits of a modular, layered, and loosely coupled application design are best



realized by implementing a services-oriented architecture using a client/server model. A services-oriented architecture provides a range of software and infrastructure services to any application component that needs them in a shared operating environment.

4.3.1 Introduction to Terms

The following terms and concepts are vital to the understanding and application of the material discussed throughout the remainder of this volume:

- **Application** – An application is a collection of software programs that execute the user interface, **business rules** (defined later), and data access operations necessary to execute a business process. It is the building block of an information system. Applications typically consist of one or more software program **modules** (defined later) that perform these operations; however, unless specific standards are followed, the implementation of an application can be unique to each programmer's particular design and development approach.
- **Application Program Interface** – An Application Program Interface (API) is a convention by which application programs may request **services** (defined later) from other software programs. Software vendors usually provide an API in order for custom-developed applications to take advantage of the special features provided by the vendor's commercial-off-the-shelf software product. However, custom-developed application programs may also employ an API in order to provide special features that can be used by other custom-developed applications. Examples of application operations that can be implemented using an API include reading or writing a database

record, checking a user's security privileges, writing data to a printer, and sending data over a network.

- **Business Rules** – Business rules are the logical specifications of the user's business requirements that help to determine the computational algorithms and operations performed against the data, which are necessary to implement a business process. A programmer must translate the business rules into programming logic, which dictates the software instructions to be executed by an application.

- **Messaging** – This is a mechanism for communicating data between application programs running on the same or on different computer platforms. When communications occur between application programs executing on different platforms, messaging uses the underlying hardware, software, and network infrastructure to send and receive the application data. Messaging can be designed and implemented to execute synchronously or asynchronously, depending upon the technology used (see Volume 5, Section 5.5, Middleware, for further details). The requirements of the business process should dictate which technique is appropriate for a given application design.

- **Database Access Middleware** – This is a set of software programs specifically designed to provide access to a database or file system in response to requests from an application program. Should the database reside on a different hardware platform from the application program needing access to the data, the database access middleware uses some form of messaging to provide the access.
- **Information Systems Group** – An information systems group is a collection of applications that operate on a common subject area database and perform business processes that are closely related to one another.
- **Logical Layers** – Logical layers represent a conceptual depiction of the primary structures within an application program for purposes of specifying and designing the program's execution features and characteristics. In order for a business application program to fully support a business process, it must contain a user interface, perform computational logic, and perform some operation against the business data. Defining the logical layers of an application provides practical boundaries for physically segmenting the application into smaller, more manageable program **modules** (see Section 4.3.5.1 for a detailed description of the logical layers defined within the Application Architecture). The interactions between logical layers of an application can be accomplished through messaging and middleware services (described later). The logical layers of an application are defined as:
    - User Interface Layer – Program operations that interact with the end-user input/output devices.
    - Business Logic Layer – Program operations that implement the computational rules and algorithms as defined by the requirements of a particular business process.
    - Data Access Layer – Program operations that perform the primitive actions against data, such as *create*, *read*, *update*, and *delete*, as implemented by the specific instructions of the data access method.
- **Modules** – Within an application program, modules are subordinate software programs that are combined with the main control program to perform specific operations when called upon by the main control program. Typically, most information systems consist of numerous application programs that repeatedly perform similar or identical computational and/or data access operations. Modules are the building blocks of an application, and several modules are usually included within a single application program. Designing application programs modularly (using modules) enables software reuse within other applications that require the same operations. Software reuse increases the maintainability of systems, improves programmer productivity, and provides a means for standard application development.
- **Services** – Services are predefined, specialized results produced from specific software programs that are designed to perform explicit data processing operations when called upon. Services can be placed into two primary categories: business application services and technical infrastructure services. Business application services modules can be designed and developed by application programmers to

provide specific computational, input/output, or data access operations when called upon by other business applications at execution time. A collection of technical infrastructure services is usually provided by the computer platform operating system, database management systems, or network platform that supports business applications. Services are necessary in order to design and implement information systems that will operate under a client/server computing model.

4.3.2 Information Systems Groups

HCFA's business, by its nature, requires access to a variety of information about Agency customers, stakeholders, and business partners, information that is derived from continuous flows of data from the Agency's business operations. The information required for the administration and management of HCFA programs is driven by the demands of the various business functions and processes performed throughout the enterprise.[3] These continuous flows of data necessitate effective and efficient data access and other capabilities, requiring an application architecture that is also information-centered.

The HCFA business functions, as defined in Volume 3 – Information Architecture[4], rely upon different kinds of business application systems for access to needed information. These systems can be grouped into three primary categories: transactional data processing systems, analytical information systems, and knowledge management information systems.

The transactional data processing system applications are needed by HCFA to maintain each of the transactional data stores defined in the Information Architecture. HCFA's transactional data stores produce very high volumes of raw data that must be well-managed. In general, transactional data processing systems manage the mountains of raw data used to run the day-to-day operations of the business.

Transactional data processing systems are designed and implemented to access and maintain transactional data stores based upon requirements of the business function processes. These applications are typified by well-defined events and responses. Examples include claims processing, attendance, and payment processing systems. They typically involve large amounts of structured data, such as relational database tables. In HCFA's environment, millions of transactions are typically processed by these systems, and the steps for each transaction are well-defined to process data quickly and efficiently.

[3] Volume 2 of the ITA, the Business Architecture, contains the current HCFA Business Function Model (BFM). As functions within the BFM are decomposed, they can be used to identify information needs.
[4] Volume 3 of the ITA, the Information Architecture, contains the current high-level subject areas and information model.

Analytical information systems are used to support program management, policy development and decision-making, and research and analytical functions within HCFA. Analytical applications perform many operations that aggregate data across different subject areas and demographic or geographic dimensions, as well as "mining" data acquired via transaction processing applications, to produce information. These operations often involve transforming, restructuring, and summarizing transactional data into subject-oriented, structured information stores that facilitate analytical activities using defined methods and techniques (see Volume 3 for a discussion of data warehousing). The functional differences between transactional data processing and analytical applications require different design and development methods, as well as different tools.

Analytical information system applications are needed by HCFA to access informational databases that are structured based upon the high-level groupings of subject areas defined in the target information model. These applications support the analytical operations that access informational databases derived from one or more transactional data stores. HCFA's transactional data stores hold tremendous volumes of data that must be transformed routinely into useful information that is accessible to the enterprise. The resulting information helps HCFA's senior leadership, program managers, business partners, and stakeholders make program and policy decisions, and it helps beneficiaries and providers make informed health care service and delivery decisions.

Knowledge management information systems are needed by HCFA to advance the capabilities of *knowledge management* as defined in the Information Architecture. Many of HCFA's business processes involve people creating, reviewing, editing, and approving documents, where a document might be a report, a model, a multimedia clip, and so on. Knowledge management applications enable people to capture, collaborate on, and share knowledge (experiences, expertise, and insights) through the creation, maintenance, and access of structured and unstructured data, as well as other forms of intellectual property. Knowledge management applications contain such features as document management and collaboration, and especially benefit from being workflow enabled. Version control, check-in/check-out, and special data-retention policies are also common requirements for these applications. The characteristics of the data accessed by knowledge management applications impose increased demands on the computing infrastructure (e.g., more bandwidth, increased CPU usage for image processing, high-end graphical display devices, increased disk storage space).

Exhibit 4-7 groups the three categories of information systems with HCFA's information needs as defined in the Information Architecture.

EXHIBIT 4-7. GROUPING OF INFORMATION SYSTEMS WITH INFORMATION NEEDS

Information Needs (IN)	Information Systems Groups
IN-1: Knowledge about beneficiary characteristics, needs, and awareness is essential, as HCFA plans to assess beneficiaries' functional status over time, conduct extensive beneficiary education programs, and reach vulnerable populations. This knowledge includes identification and information collection regarding special populations, such as immigrants.	Transactional Data Processing Systems Analytical Information Systems
IN-2: Comparative data, benchmark and quality indicators, and outcome-oriented measures are crucial to HCFA's ability to ensure that beneficiaries have access to new technologies and medical practices as they emerge and are supported by authoritative scientific evidence.	Transactional Data Processing Systems Analytical Information Systems
IN-3: Cost and financial data, such as that related to policies and programs, interventions and outcomes, and health care service delivery, is key to evaluating health plan financing options and overall health system expenditure trends.	Transactional Data Processing Systems Analytical Information Systems
IN-4: Outcome and assessment data is crucial to HCFA's ability to evaluate different service delivery models, specific intervention strategies, population and setting trends, and the impact of program changes on various populations.	Transactional Data Processing Systems Analytical Information Systems
IN-5: Knowledge of customer expectations and satisfaction is necessary to assess the ongoing effectiveness of HCFA programs and the improvement of beneficiary health.	Knowledge Management Information Systems
IN-6: Integrated health information, from a variety of sources, supported by common data exchange standards is necessary to: • Provide consumer information on health care plan and provider options, including comparative data on services, costs, and	Analytical Information Systems Knowledge Management Information Systems

Information Needs (IN)	Information Systems Groups
quality of care. • Support comparative population and practice studies that require connection of service providers and patient episodes, or a combination of clinical, survey, and other types of data. • Support analysis across Medicare, Medicaid, and Child Health populations.	
IN-7: User-empowering, Internet-based access to HCFA's information assets is necessary to ensure that Agency personnel have easy and immediate access to required information.	Analytical Information Systems
IN-8: Workforce skills, training, and satisfaction; required competencies; and knowledge of industry trends and developments in the human resources management and management science fields are key to the development and maintenance of an effective, customer-focused team.	Knowledge Management Information Systems
IN-9: Knowledge of IT trends and best practices is necessary to enable effective management of IT investment contracts and to ensure the appropriate use of technology.	Knowledge Management Information Systems

Exhibit 4-8 lists the logical applications within each information systems group defined for the target Application Architecture. Each information systems group relates information objects identified in the Information Model presented in the Information Architecture. The Information Model identifies and relates HCFA information subject areas to the transactional data processing, analytical information, and knowledge management information needs of HCFA's business functions. Collectively, these logical information systems groups establish the conceptual applications needed to satisfy HCFA's data and information access needs. Note, however, that the conceptual applications may differ from the actual physical applications identified within HCFA's current systems inventory.

EXHIBIT 4-8. INFORMATION SYSTEMS GROUPS

Information Systems Groups	Description
Transactional Data Processing Systems	
Beneficiary Data Management System	Applications that maintain access to data about HCFA program beneficiaries.
Provider Data Management System	Applications that maintain access to data about health care service providers.
Insurer Data Management System	Applications that maintain access to data about insurance partners involved in HCFA programs.
Health Care Plan Data Management System	Applications that maintain access to data about benefits packages and cost schedules for health insurance programs and plans.
Utilization Data Management System	Applications that maintain access to data about services provided to beneficiaries.
ESRD Data Management System	Applications that maintain access to data about the care provided to beneficiaries with kidney disease.
Survey Data Management System	Applications that maintain access to data about provider-quality surveys.
General Ledger Data Management System	Applications that maintain access to data about financial accounting of Medicare Trust Funds.

Information Systems Groups	Description
Analytical Information Systems	
Business Management Information System	Applications that provide access to information about agreements, business statements, policy & regulations, legal documents, and material resources.
Health Care Finance Information System	Applications that provide access to information about financial statements and health plans.
Health Care Stakeholder Information System	Applications that provide access to information about persons and organizations, such as clients, partners, insurers, providers, and State Medicaid agencies.
Health Care Assessment Information System	Applications that provide access to information about provider assessments and quality surveys.
Health Care Services Information System	Applications that provide access to information about services and disputes & resolutions.
Geographic Locations Information System	Applications that provide access to information about location elements and location groups within the HCFA enterprise.
Knowledge Management Information Systems	
Document/Collaboration Information System	Applications that provide and control access to the inventory of HCFA documents and records.
Human Resources Information System	Applications that provide access to information about HCFA personnel skills, expertise, education, and training, as well as contractor capabilities.

Attachment A provides a matrix that maps the relationships between the information systems groups and the HCFA business functions they support. Attachment A also provides matrices that associate HCFA's subject area databases and physical applications with the information systems groups. Most of the physical legacy application systems predate the ITA. Consequently, gaps and overlaps inevitably will exist between the physical legacy applications and the previously defined information systems groups. Future analysis of the legacy systems

will permit HCFA's portfolio, through strategic application development planning, to evolve consistent with current and future information needs.

4.3.3 Application Boundaries

An important principle in designing information systems to be consistent with the target environment is the separation of the logical data operations into discrete applications (see Section 4.4.1, Design Principles). For example, information systems should not be designed to perform both transactional data processing and analytical operations within a single application. These distinct information systems groups perform very different logical operations against data, and have very different data access requirements. They also impose very different demands on the computing infrastructure. Since the processing performed by these systems differs significantly, the design, development, and deployment considerations for the applications within each information systems group also differ.

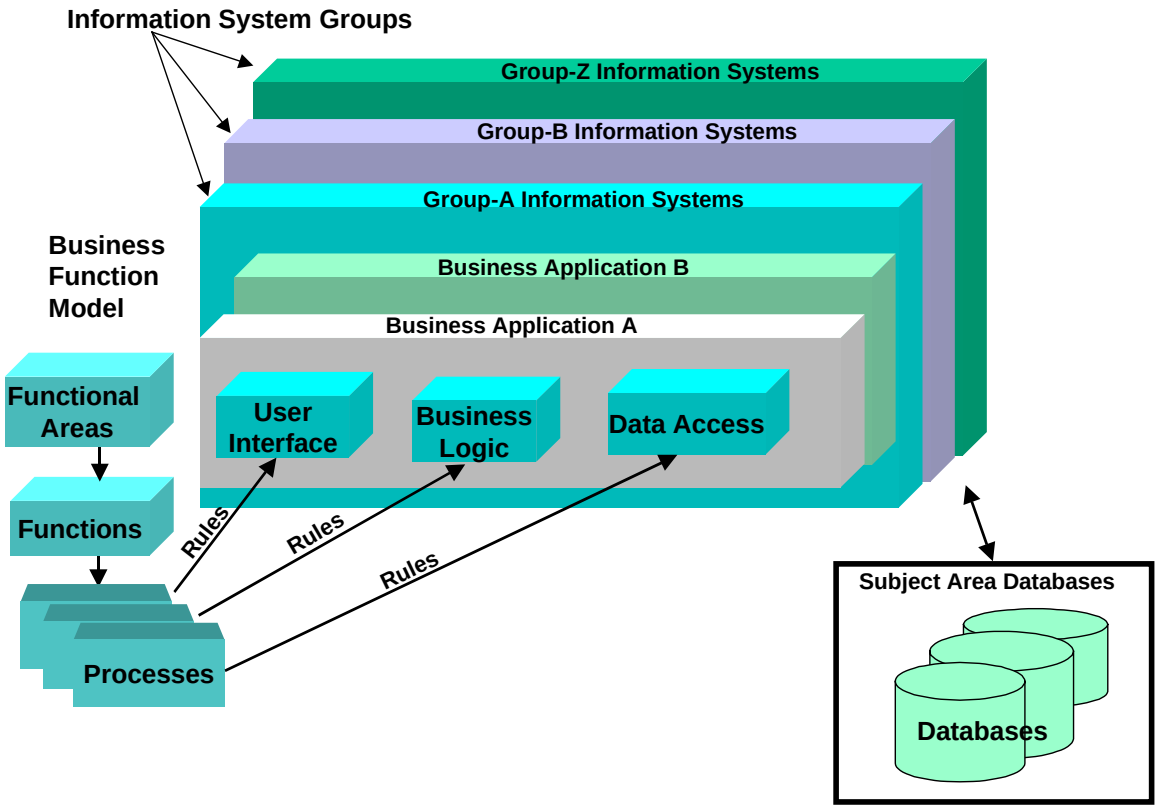
Applications designed and developed for the target environment need to be adaptable and maintainable. Designing applications to support multiple business processes usually requires accessing different kinds of data and/or information stores, which leads to the development of complex monolithic systems that are difficult to modify and maintain. To minimize this problem in the target environment, business applications should be designed to mimic a discrete business process. This will simplify each application's logic and restrict its size and scope (see Section 4.4.1, Design Principles).

Limiting the size and scope of an application necessitates clear design boundaries. Appropriate boundaries for business applications need to be established so that systems designers have a clear understanding of what is within and what is beyond the scope of each business process. The logical boundaries of an application are best determined by the scope of the business process that it supports. HCFA's business processes are defined by decomposing the functions identified in the HCFA Business Function Model (BFM). Each business function within the BFM can be decomposed into a set of discrete business processes, and each business process is determined by a set of business rules, based upon the requirements of HCFA's business managers, customers, and stakeholders. Once a function within the BFM has been decomposed, the boundaries for the application can be delineated, and alignment is possible. Exhibit 4-9 provides a conceptual overview of how this alignment might occur.

Business applications are typically constructed from many software modules containing logic that provides such services as the user interface, business processing, and access to databases based upon business rules. Business rules capture the behaviors and events associated with a business process. Business process requirements determine the business rules that primarily influence the design of an application. Requirements analysis is used to identify those parts of a business process that present opportunities for automation.

Tacitly, then, the logical boundaries of an application should extend to the boundaries of the business process that it supports. However, the details of each discrete business process can vary widely, and not all aspects are necessarily automated. Thus, the boundaries for a given business process cannot be defined using a formula or cookie-cutter approach. Thorough requirements analysis, completed early in the SDLC, is the only effective way of determining and documenting the details of each business process.

EXHIBIT 4-9. ALIGNMENT OF BUSINESS PROCESSES AND APPLICATIONS WITHIN HCFA'S INFORMATION SYSTEMS GROUPS



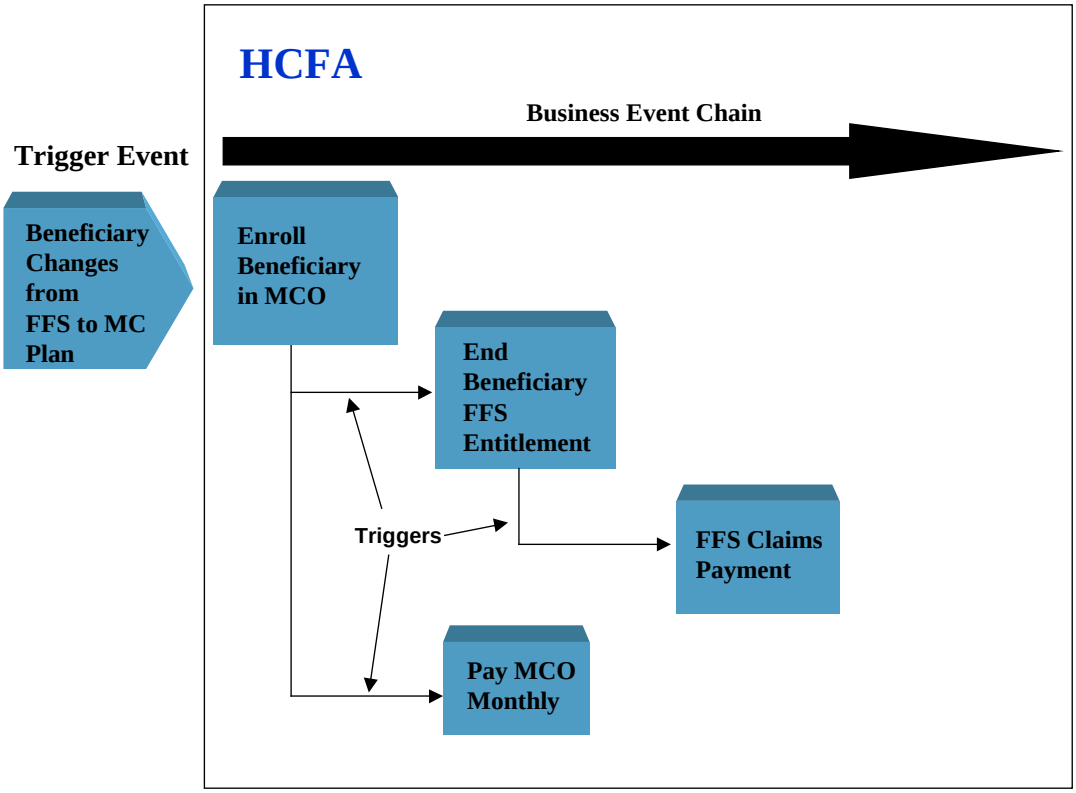
System development methods vary; therefore, the methods and procedures for performing requirements analysis depend upon standards chosen by HCFA (see Section 4.6). Regardless of the method used, the salient outcomes of requirements analysis that are necessary for setting appropriate application boundaries include:

- *Identifying the Trigger Event* – A trigger event is one or more external actions by a customer or business partner that cause a business process within HCFA to be initiated. A single business process could be invoked by one or more trigger events. Another business process could also be a trigger event.

- *Detailing the Activities, Tasks, and Steps* – Specific manual and automated activities, tasks, and steps make up a business process. Activities describe the actions to be taken, tasks describe the roles and responsibilities of those involved, and steps describe the sequence and dependencies of activities and tasks.
- *Identifying Predecessor and Successor Processes* – Completing one business process might require initiating another process. HCFA's business rules determine the sequence, dependency (predecessor and successor), and interrelationships between business processes. Usually, the success or failure of a predecessor business process determines which (if any) successor processes are to be initiated.

To illustrate how the requirements process analysis might work, Exhibit 4-10 depicts a typical business process interaction within HCFA. The scenario is that of a Medicare beneficiary electing to change enrollment from traditional fee-for-service (FFS) to insurance coverage offered by a Managed Care Organization (MCO), which is the trigger event. (The scenario depicted is for illustrative purposes only, and may not accurately reflect actual HCFA business rules or processes.)

EXHIBIT 4-10. BUSINESS PROCESS INTERACTIONS



In this scenario, four discrete business processes are required, each with its own unique business rules determining the procedures that must be performed by HCFA because of the trigger event:

(1) The Enroll Beneficiary in MCO process has a set of business rules and procedures for updating its database to enroll the beneficiary in the elected MCO. Included in these business rules is the requirement to notify the Medicare FFS Entitlement process to end the beneficiary's FFS insurance coverage entitlement.

(2) Upon notification, the End Beneficiary FFS Entitlement process has business rules to update the database to end the beneficiary's FFS entitlement, and then to notify the FFS Claims Payment process of the event.

(3) Upon notification, the FFS Claims Payment process has business rules to update its database to deny future payments for FFS claims received for the beneficiary from providers.

(4) Finally, the Enroll Beneficiary in MCO process must update its database to increase the number of beneficiaries enrolled in the particular MCO elected by the beneficiary so that the Pay MCO Monthly process calculates proper monthly payments to MCOs.

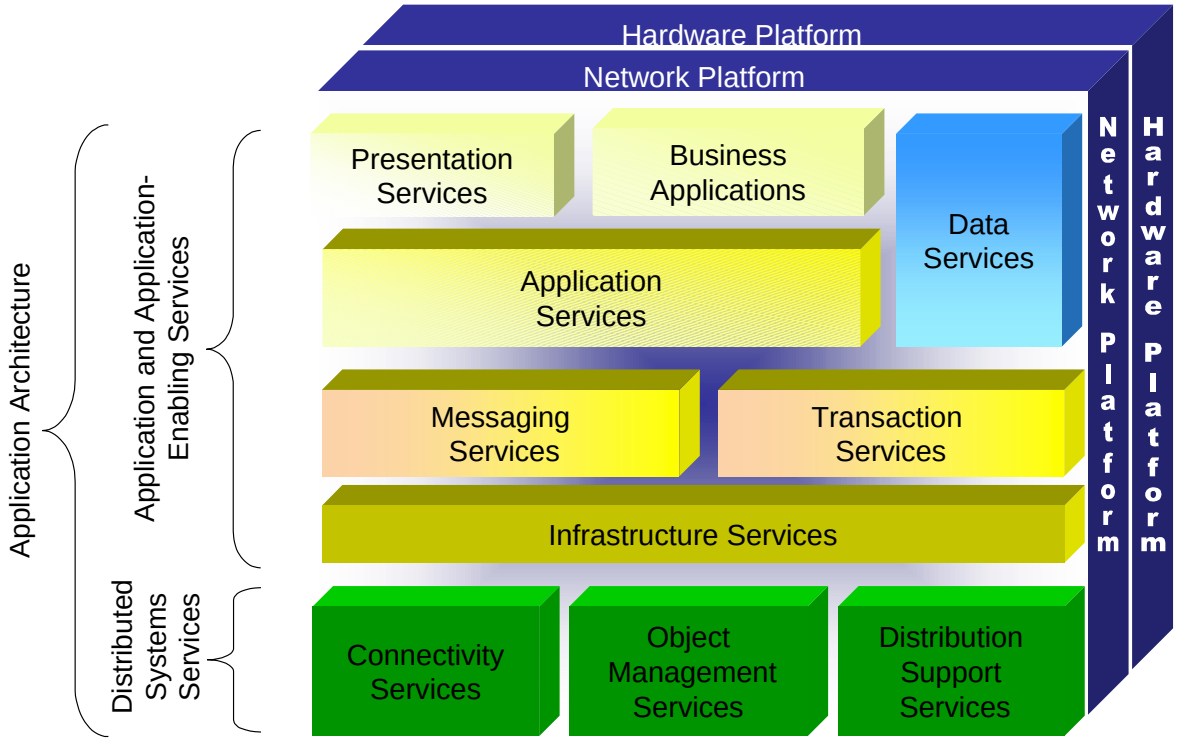
In the final analysis, defining logical application boundaries that mimic a business process is a matter of delineating where each business process starts and stops. Predecessor and successor processes may have specific information needs that determine interface requirements between the logical application. Standards define the format and structure of information interfaces between application boundaries.

4.3.4 Application Architecture Services Model

This section presents a high-level view of the Application Architecture and describes its technical components and services. These services represent modules that provide functional capabilities or common system services. They are the building blocks for developing applications that are consistent with HCFA's target Application Architecture. For a more detailed discussion of these services, see Attachment B.

The Application Architecture services are shown in Exhibit 4-11. This diagram displays both the Application Architecture component services (on the front pane of the cube) and the infrastructure architecture elements (layers of the cube behind the application services). This volume addresses the roles and relationships among the various component services. The standards associated with these services, and the Infrastructure Architecture elements, are discussed in Volume 5.

EXHIBIT 4-11. APPLICATION ARCHITECTURE SERVICES MODEL



The value of the Application Architecture Services Model is that it provides the structure upon which HCFA can develop business applications that are modular and portable. This model depicts how application services insulate the business application from platform-level and even system-level services, allowing for increased portability.

Ideally, the design of business applications would be insulated from the lower platform-level and system-level services. Unfortunately, this may not be possible in all instances. In cases where insulation is not possible, it is important to understand and consider the impact of coupling business applications to specific system services or platforms. For example, some relational database management systems offer a complete application design and development environment. From a development perspective, this means that the user interface, business logic, and data access can all be handled through a single suite of tools. While this appears to provide some advantage, there is a trade-off in portability and modularity when using some of the current technology tool suites. This type of environment generally involves the use of proprietary procedural languages for the development of user interface screens and business logic. It is relatively easy to become locked into a vendor's proprietary technology environment.

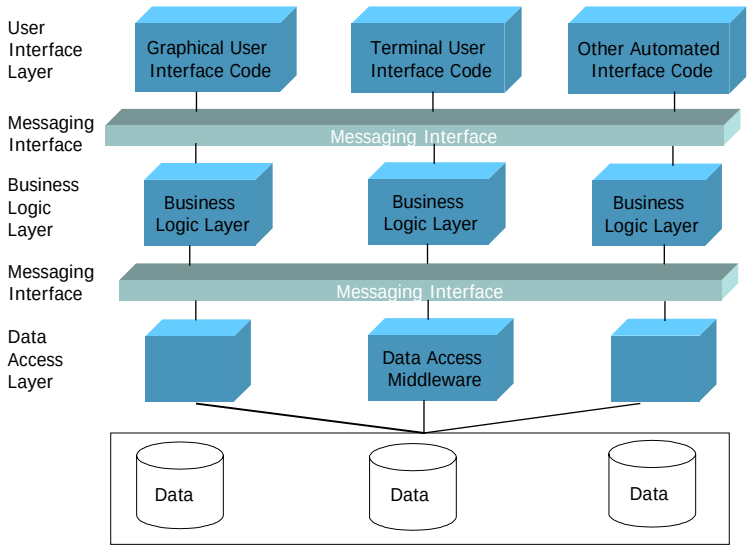
4.3.5 Application Design Concepts

The client/server model of computing encourages more modular approaches to application design in order to promote sharing, interoperability, and reuse. The Application Architecture Services Model discussed previously provides a framework for designing information systems using the client/server model. A concept of layering the logical design of an application is also used to provide a consistent framework for implementing the services comprising the model. By layering the application design, client program and server program operations and the services they use are isolated to the parts of the business process it best supports. Implementing a physical design corresponding to the logical layers of an application permits deploying and operating client and server programs on different distributed computing platforms, as described in Section 4.3.6.

4.3.5.1 Logical Layers

A common approach to developing client/server applications is to divide the application into three logical layers: a user interface layer, a business logic layer, and a data access layer. Like features, operations and services are organized into these layers. Dividing applications into layers provides several benefits. It permits layers to be sized and scoped appropriately, and reduces the complexity inherent in monolithic (stovepipe) solutions. The resultant applications are more flexible, portable, and easier to maintain because program segments are smaller and complexity is reduced. From a configuration management perspective, it is much easier to enforce stricter standards for smaller application programs. Moreover, smaller programs that perform specific operations are more likely to be candidates for reuse across the enterprise. Exhibit 4-12 provides a conceptual view of the logical application layers, including the messaging and data access middleware interfaces.

EXHIBIT 4-12. LOGICAL APPLICATION LAYERS



The logical layers of an application are defined as follows:

- **User Interface** – The layer of an application that interacts directly with end-user input/output devices (e.g., Windows workstations or a printer/fax device). It encompasses a variety of operations, such as window or screen management, keyboard and mouse handling, end-user help functions, general input editing and data type validation, or formatting data for output to a laser printer or plotter device. The user interface layer is the most visible, and arguably the most important aspect of the business process supporting the end user.
- **Business Logic** – The layer of an application that implements the particular requirements of a business process. Typical operations at this layer consist of controlling the logical flow of interaction between the end user (via the user interface layer), access and manipulation of data or information (via the data access layer), and specific computational algorithms to be performed.
- **Data Access** – The layer of an application that includes the operations needed to store, access, and maintain data or information as necessary to support a business process. The data accessed within this layer can include both structured and unstructured formats, depending upon the application requirements. For the most part, a commercial relational database management (RDBMS) or proprietary file access system provides the services performed within this layer.

The division of applications into logical layers and the inherent physical program design characteristics necessitate services that enable communication between logical and physical layers. Two different mechanisms provide these services: messaging and data access middleware, as described below.

- **Messaging** – This layer provides a mechanism for application programs within the user interface layer or the business logic layer to communicate with each other and between layers, under a client/server model. Messaging is typically provided to an application as a service through a standard interface. The messaging service can include synchronous or asynchronous delivery, depending upon the implementation. Both types of messaging service may utilize the underlying network transport protocols to provide services in a distributed computing environment. An important feature that messaging should provide is location transparency. That is, an application program (user interface, business logic, or data access) in a client/server environment should be able to consistently execute the service independent of the computer platform upon which the client or server resides. The messaging service uses other system services to locate and transport messages to their proper destination.

- **Data Access Middleware**[5] – This provides a standardized mechanism for accessing data sources maintained within the data access layer of an application. Data access middleware is typically more sophisticated than messaging software. For example, in addition to the services that messaging provides, data access middleware may provide "translation" services for accessing legacy data, or a service that allocates local storage to hold the results of the query. Typically, commercial software products specifically designed to operate against a particular RDBMS or proprietary file structure provide data access middleware services.

4.3.5.2 Advantages of a Layered Application Design

Separating an application into discrete layers, as discussed in the last section, permits application services to be scaled and positioned where appropriate and reduces the complexity inherent in single-platform solutions. Client/server computing allows specialized components to mix-and-match to achieve the best results. The client/server model has the following key characteristics:

- The client and server can interact seamlessly.
- The client and server are generally located on separate platforms and are connected by a network. (This does not preclude having both the client and server on the same platform.)
- The client can be changed without affecting the server, and the server can be changed without affecting the client.
- The client and server functions are independent and may perform interchangeable roles.
- The server can serve multiple clients concurrently, and, conversely, a client can access multiple servers.

4.3.5.3 Application Program Interfaces

Before discussing the individual application system design models, it is important to discuss the role of APIs in the development of applications.

An API is the convention by which one application service requests services from another. An API is a way of requesting services such as file transfer and security from the underlying system of hardware, software, and networks. APIs insulate application programmers from complexity and greatly reduce the amount of time required to develop new applications or modify existing ones.

The environment manager, business applications, infrastructure services, and application services components use APIs to communicate with each of the various system services. APIs

[5] Data access middleware technology and standards are addressed in HCFA ITA Volume 5 – Infrastructure Architecture, Section 5.5, Middleware.

should be standardized to allow for portability. Proprietary extensions to APIs should be avoided unless dictated by business needs. Section 4.3.6 describes the information systems design models and details the unique features and characteristics of each.

4.3.5.4 Services and API Standards

A sound IT architecture is based upon approved standards that support the business requirements and maximize the interoperability of information systems across the enterprise. National and international standards bodies, and the standards associated with each technology service identified in the HCFA Technical Reference Model, are discussed in Volume 5. The APIs adopted for use within HCFA will be a combination of public, de facto, and HCFA internal standards. To ensure that our standards-based approach fully supports the needs of our application environment, Application Architecture workgroups made up of members from across HCFA will review candidate standards and recommend specific standards to be included in Volume 5.

With a comprehensive strategy, HCFA can maximize benefits and minimize risks by using existing standards wherever they meet requirements. As business requirements change, software innovation is often required. Innovation, however, inevitability extends beyond standards. This is often the case when the requirements include connecting to existing proprietary systems. Compromises and deviations from adopted standards must be the result of a business decision based upon the availability of technologies, and must be balanced with the risk of using proprietary approaches.

Business requirements will occasionally exceed the capabilities of commercially available components. In some instances, a business requirement will dictate the use of non-standard components and API sets. In these cases, there are essentially two viable approaches. Each of the following approaches requires that appropriate APIs be designed and constructed to support the approach. The two approaches are:

- Functional compensation – The use of software provided to fill a gap in functionality between an architecture API and a given product.
- API translation – Processes by which a service request made via one API is translated into one or more service requests in another API, one that is understood by the underlying product. It could be as simple as translating a return code from one value to another, or as complex as reformatting a single request into multiple requests.

4.3.6 Information Systems Design Models

The Application Architecture Services Model described in Section 4.3.4 provides a framework for developing adaptable information systems using a client/server model of computing that can operate in a distributed environment. In a client/server model of computing, clients are programs that request services to be performed by other programs, and servers are programs that respond to these requests for services. Designing information systems using a client/server approach permits software programs to cooperatively satisfy business needs by requesting services of each other regardless of whether the software programs reside on the same or on

different platforms. However, the client and server programs must be properly configured for deployment onto the computing infrastructure in order for successful interoperability to be achieved. The use of standards promotes interoperability. Without standards, disparities in the deployed infrastructure are unavoidable, which complicates systems design efforts, makes systems management difficult, and drives up HCFA's operating costs.

HCFA must be able to deploy its infrastructure components cost-effectively and efficiently in order to support the needs of Agency information systems. Application developers must be enabled to deliver applications quickly and efficiently to satisfy the business requirements. Additional guidance is needed for configuring and deploying the technical infrastructure[6] to support distributed application systems. Establishing standard models for the design of client/server information systems also helps to determine the standards for technical infrastructure configurations. The systems design model concepts described in the following sections provide the basis for achieving this standardizing.

4.3.6.1 Systems Design Model Assumptions

A careful feasibility assessment of using various client/server computing scenarios within HCFA's environment produced eight alternatives that are considered viable systems design models. The following assumptions formed the basis for selecting these models:

- In the early stage of application delivery, the emphasis should be on specifying the design of application processes that implement the business requirements. The use of models provides a mechanism for developing a logical, high-level specification of application processes, before delving into technical details.
- As HCFA's application development methodology evolves toward model-driven development to enhance application delivery efficiency, the modeling objects can be based upon these model concepts.
- Only a finite set of client/server computing models makes sense for HCFA's business operations environment; therefore, only a few design models are necessary to help standardize the design of systems. The eight models described later in this volume are intended for use in application development projects to help describe requirements for the computing infrastructure based upon standards, rather than project-specific infrastructure.
- Uniqueness among HCFA's information systems is less important, except with regard to the user interface, process-specific business rules, and the data necessary to support a particular business process. All other aspects of a system's infrastructure design (i.e., operating system, network, middleware, etc.) can benefit from standardization, with little impact from a system-user perspective.
- Systems configuration management and capacity planning are disciplines within the domain of infrastructure management that are addressed at later stages in the system

[6] Technology standards are described in Volume 5 – Infrastructure Architecture.

development life cycle. These disciplines are concerned with provisioning sufficient infrastructure resources to accommodate the operational needs of information systems. No attempt is made to address configuration management and capacity planning within the Application Architecture or the design models.

4.3.6.2 Overview of Systems Design Model Concepts

Each systems design model represents a logical client/server computing configuration that can be used for different application system implementation schemes. Each model is unique in the placement of application services provided by the user interface, business logic, and data access layers of an information system across the computer platform infrastructure (see Section 4.3.5.1, Logical Layers).

The ***user interface layer*** handles all interactions between input/output devices and the business application system. Input/output devices can range from "dumb" character-based terminals and printers to sophisticated graphical or alternative media devices. The user interface layer includes the following services:

- Presentation Services – Input/output presentation services
- Infrastructure Services – User logon profile, presentation, and interface support services
- Transaction Services – Transaction initiation and termination services

The ***business logic layer*** includes the business rule logic as developed to implement an application process. The complexity of business rule logic can range from simple modules that perform discrete processing operations, to sophisticated modules that are distributed across computer platforms that operate cooperatively. The business logic layer includes the following services:

- Business Application Logic – Process-specific business rule modules (logic, data consistency edits, computations, I/O formatting).
- Application Services – Common business rule, edit, computation, and format modules, and object class services.
- Transaction Services – Execution, flow control, and termination services.
- Infrastructure Services – User access profile, security, call-level interface, and computer platform-dependent input/output services.

The ***data access layer*** includes all aspects of data management for workgroup or enterprise database access. The data management operations (create, read, update, and delete) may utilize sequential flat file structures, proprietary file structures, or a sophisticated RDBMS. A basic tenet of enterprise data management is that no data of enterprise importance should be maintained on the client platform (see Volume 3, Information Architecture). The data access layer includes the following services:

- DBMS or File Access – Access to information system databases and other file structures.
- Metadata – Access to data about data, as well as standard reference databases.
- Data Translation – Access to data formatting, translation, and movement services.
- Data Integrity – Support for database backup and recovery services.
- Middleware – Standard interfaces to database and file access services.

In a distributed environment, the user interface, business logic, and data access layers of the design models require a ***messaging layer*** to provide the services necessary for intra- and interprocess interoperability. The messaging layer enables the workgroup server platforms to interconnect with the enterprise server platforms over a wide area network, and the client platforms to interconnect with workgroup server platforms over a local area network. The messaging layer includes the following services:

- Interprocess Messaging Services – Interfaces to asynchronous message queuing and synchronous messaging.
- Distributed Object Messaging Services – Interfaces to ORB middleware.

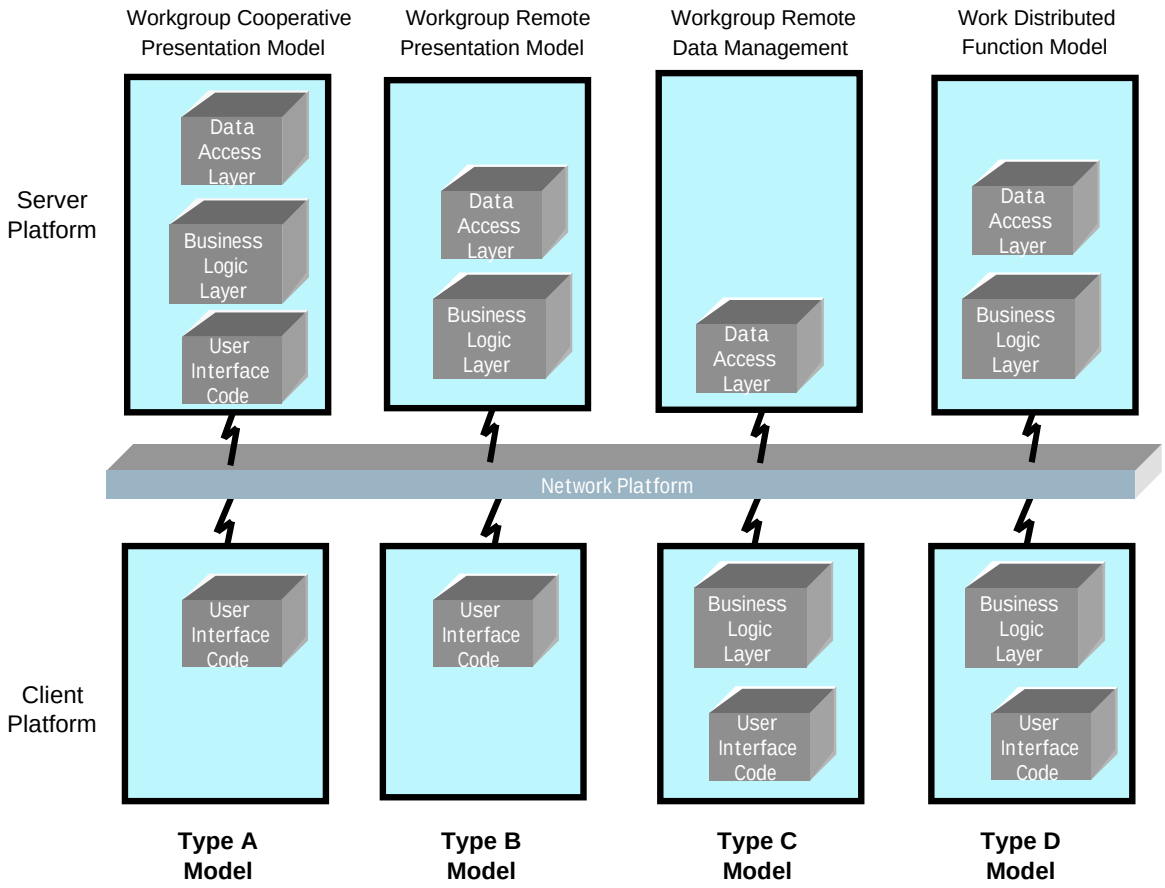
The application services mentioned above represent the minimum features needed to implement a client/server system, and may not be all-inclusive. Additional services may be identified for the various layers within the models. The models are intended as a tool for use by application developers early in the system design stage to convey business requirements for the infrastructure, without directly specifying technology. Each systems design model has a corresponding infrastructure configuration template that identifies the technology standards for deploying the target infrastructure architecture.

Some information systems are designed for use in performing business function processes that are isolated to specific HCFA organizational units or functional workgroups, while others support business function processes that are performed throughout the enterprise. Applications generally used by workgroups, for example, access data that is isolated to specific business function operations. Enterprise-wide business functions, on the other hand, are carried out by multiple divisions or workgroups that are geographically dispersed, often requiring more processing-intensive applications that access information or data stores of greater volume.

4.3.6.3 Workgroup Information Systems Design Models

Four design models are provided for use in designing workgroup information systems that support organizational components or functional units within HCFA's environment. Exhibit 4-13 graphically depicts these models.

EXHIBIT 4-13. WORKGROUP INFORMATION SYSTEMS DESIGN MODELS



Following is a brief description of each model:

- **Workgroup Cooperative Presentation Model (Type A)** – This design model can be used to place a minimum number of presentation services on the client platform and all other services on the workgroup server platform. This is sometimes referred to as a "thin client" model because it demands the least performance of the client platform. The user interface layer is loosely coupled and uses cooperative processing between the client and server platforms to provide the presentation services.
- **Workgroup Remote Presentation Model (Type B)** – This design model can be used to place all presentation services on the client platform, while all other application

services are placed on the workgroup server platform. This model is sometimes referred to as an "application server" because the server software originating from one source (custom-developed and/or COTS/GOTS products) can be used with client presentation software originating from different sources.

- **Workgroup Remote Data Management Model (Type C)** – This design model can be used to place all database management services on the workgroup server platform. This model is sometimes referred to as a "database server" because client application software originating from different sources (custom-developed, COTS/GOTS products, or a combination) that perform similar database operations can be used for remote database access.
- **Workgroup Distributed Function Model (Type D)** – This model can be used to target the placement of selected application services within the business logic layer on either the client platform or the workgroup server platform. (Note, however, that database management services should never be placed on the client platform.) Application software originating from the same source (either custom-developed or COTS/GOTS products) is typically used because of the integration of application programs in the business logic layer.

Exhibit 4-14 presents a side-by-side comparison of the salient features and characteristics supported by the four workgroup information systems design models.

EXHIBIT 4-14. WORKGROUP DESIGN MODEL FEATURES AND CHARACTERISTICS

Model Features and Characteristics	Type A	Type B	Type C	Type D
1. Workgroup information system users can share access to a local subject area database, exclusive of other workgroups	✔	✔	✔	✔
2. The workgroup information system can operate independent of other workgroup server applications (no interprocess dependencies exist)	✔	✔	✔	✔
3. Databases can be hosted entirely on the workgroup server platform, or partitioned such that no cross-platform database access is needed	✔	✔	✔	✔
4. Metadata can be co-located and maintained with the databases on the workgroup server platform	✔	✔	✔	✔

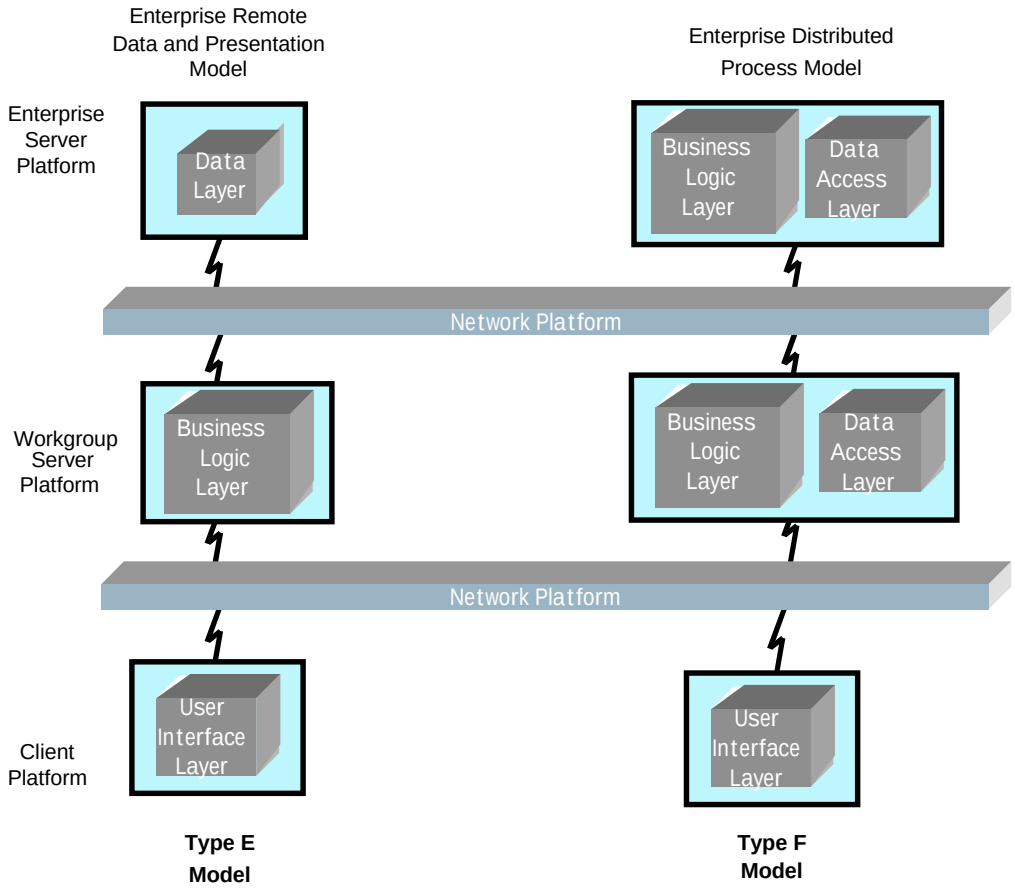
Model Features and Characteristics	Type A	Type B	Type C	Type D
5. Database security can be fully managed on the workgroup server platform	✔	✔	✔	✔
6. The business logic layer can be placed on the workgroup server platform	✔	✔		✔
7. The business logic layer can be placed on the client platform			✔	✔
8. The business logic layer can handle data consistency edits on the workgroup server platform	✔	✔		✔
9. The business logic layer can handle data consistency edits on the client platform			✔	
10. The business logic layer can handle input/output data formatting for presentation to the user interface			✔	✔
11. The user interface layer can be placed on the workgroup server platform	✔			
12. The user interface layer can be placed on the client platform	✔	✔	✔	✔

Multiple physical instances of a model can be implemented to support workgroup users who are clustered at different business operating locations within the HCFA enterprise.

4.3.6.4 Enterprise Information Systems Design Models

The following four design models are provided for use in designing large-scale information systems that support business function processes requiring enterprise-wide data access. Exhibit 4-15 and Exhibit 4-16 graphically portray these models, and each is followed by a brief description of the associated models.

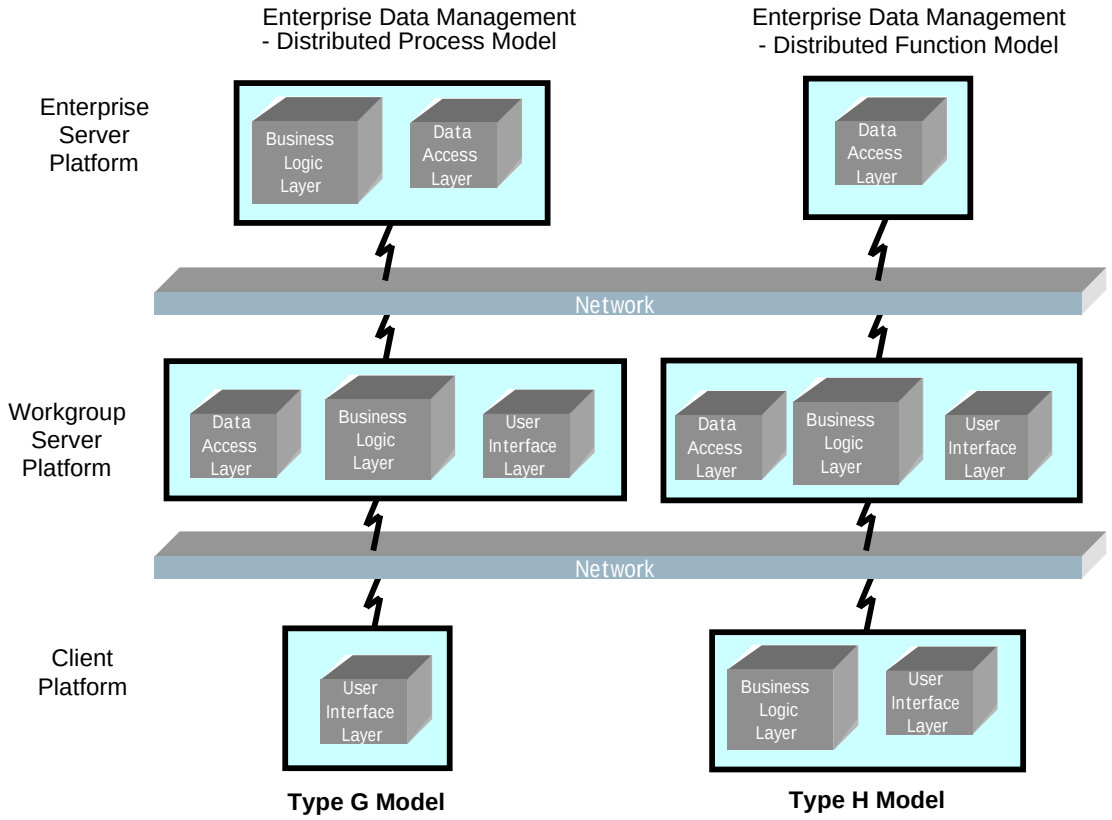
EXHIBIT 4-15. ENTERPRISE INFORMATION SYSTEMS DESIGN MODELS



- **Enterprise Remote Data and Presentation Model (Type E)** – This model can be used to place the services of each logical layer of an application onto a separate platform in a classic three-tiered client/server implementation. The enterprise database can be managed centrally and accessed remotely by one or more workgroups in different locations throughout the enterprise. Workgroup application software that is supplied from different sources (either custom-developed or COTS/GOTS products) can be utilized to access the remote database across platforms by using standardized interfaces.
- **Enterprise Distributed Process Model (Type F)** – This model can be used to partition the business logic layer for placement onto separate platforms closest to an enterprise database and the workgroup databases. Business processes with specific requirements for managing access to enterprise databases and workgroup databases can be supported by targeting the placement of the business logic layer onto the

appropriate server platform. The application software used on the different platforms would most likely originate from the same source (either custom-developed or COTS/GOTS products) because of the integration between the business logic and data access layers.

EXHIBIT 4-16. ENTERPRISE DATA MANAGEMENT SYSTEMS DESIGN MODELS



- **Enterprise Data Management and Distributed Process Model (Type G)** –This model can be used to partition the business logic layer for placement onto separate platforms closest to an enterprise database and the workgroup databases. Business processes with specific requirements for managing access to enterprise databases and workgroup databases can be supported by targeting the placement of the business logic layer onto the appropriate server platform. With this model, the user interface layer can also be partitioned for placement onto the workgroup server and client platforms. This allows workgroup users to share the use of specialized input/output devices (such as plotters, faxes, audio response units, etc.) by placing custom presentation services onto the workgroup server platform. The application software used on the different platforms would most likely originate from the same source (either

custom-developed or COTS/GOTS products) because of the complex integration between layers.

- **Enterprise Data Management and Distributed Function Model (Type H)** – This model can be used to partition the business logic layer for placement onto separate platforms closest to the workgroup databases and client platforms. Business processes needing standard access to an enterprise database, but also requiring specific access to workgroup databases, can be supported by partitioning and targeting the placement of business logic layer operations onto the workgroup server platform. With this model, the business logic and user interface layers can also be partitioned for placement onto the workgroup server and client platforms. This allows workgroup users to share the use of specialized input/output devices (plotters, faxes, audio response units, etc.) by placing custom presentation services onto the workgroup server platform. The application software used on the workgroup server and client platforms would most likely originate from the same source (either custom-developed or COTS/GOTS products) because of the complex integration between layers. However, the enterprise database access software could be implemented as a "database server".

Exhibit 4-17 presents a side-by-side comparison of the salient features and characteristics supported by the four design models for enterprise information systems.

EXHIBIT 4-17. ENTERPRISE DESIGN MODEL FEATURES AND CHARACTERISTICS

Model Features and Characteristics	Type E	Type F	Type G	Type H
1. Enterprise information system users can share access to subject area databases that are accessible throughout the enterprise	✔	✔	✔	✔
2. Databases can be hosted entirely on a single enterprise server platform	✔			
3. Databases can be partitioned between enterprise server and workgroup server platforms (cross-platform data access is required)		✔	✔	✔
4. Workgroup users can share access to a local subject area database (partition) exclusive of other workgroups		✔	✔	✔
5. Metadata can be co-located and maintained with subject databases residing on the enterprise server platform	✔	✔	✔	✔
6. Metadata can be co-located and maintained with subject databases residing on the workgroup server platform		✔	✔	✔

Model Features and Characteristics	Type E	Type F	Type G	Type H
7. Database security can be managed fully within the enterprise server platform	✔			
8. Database security can be managed to encompass the enterprise server and workgroup server platforms		✔	✔	✔
9. The data access layer can be placed on the enterprise server platform	✔	✔	✔	✔
10. The data access layer can be placed on the workgroup server platform		✔	✔	✔
11. The business logic layer can be placed on the enterprise server platform		✔	✔	
12. The business logic layer can be placed on the workgroup server	✔	✔	✔	✔
13. The business logic layer can be placed on the client platform				✔
14. The user interface layer can be placed on the workgroup server platform			✔	✔
15. The user interface layer can be placed on the client platform	✔	✔	✔	✔

Managing large data volumes and supporting processing-intensive information systems require a computer, network, and storage platform infrastructure equal to the task. These infrastructure resource requirements are determined by many factors that are unique to the business process and operational environment. The factors to consider in determining an information system's infrastructure resource requirements include:

- Data volume, frequency, and volatility.
- Data and application security and integrity requirements.
- Geographical dispersion of users.
- Data sharing and access requirements.
- Use and integration of COTS/GOTS applications versus custom-developed software.

Systems analysts and designers must consider these and other factors (such as cost constraints) in determining an optimal information system design.

4.3.7 Development and Integration Considerations

Business requirements are satisfied by application solutions that are delivered in one of three methods or alternatives: custom applications, COTS/GOTS applications, and modified COTS/GOTS applications. These three alternatives add a dimension to the application type that affects the design and deployment template used and the underlying application services required. These alternatives are discussed further in Sections 4.3.7.1, 4.3.7.2, and 4.3.7.3, respectively.

Regardless of the application type, any application that does not adhere to the standards and design principles defined for the target Application Architecture is considered to be a *legacy* application. Legacy applications must be replaced or cost-effectively integrated into the Agency's ITA so that the business processes they support can evolve and improve over time. The Application Architecture model supports the use of new techniques that provide interaction between legacy applications and data and new applications. The ability to use these newer techniques to integrate legacy applications into the ITA depends to a great extent upon the degree to which changes can be made to each application. Decisions to do so must rely upon a case-by-case analysis of each legacy application as new business and technology needs or opportunities are identified.

4.3.7.1 Custom Applications

Custom applications are those applications developed and maintained wholly by the HCFA development staff or by direct contract support. These applications are generally designed to support the unique mission requirements of HCFA that cannot be supported by commercially available software packages. All newly developed custom applications of any type must adhere to the policies and standards of the ITA. The degree to which legacy custom applications can be integrated into the architecture will depend upon the source code language, application structure, application platform, and data structures involved. Decisions to integrate a legacy application or to reengineer the application entirely must be made on a case-by-case basis.

4.3.7.2 COTS/GOTS Applications

Commercial-off-the-shelf (COTS) and government-off-the-shelf (GOTS) applications support a wide range of business functionality commonly required in both the business and Federal sectors. As identified in HCFA's IT Guiding Principles, COTS or GOTS software should be used in lieu of custom-developed software if mission requirements can be met and the COTS/GOTS software meets HCFA's architectural requirements. Because of the rigid nature of most legacy COTS/GOTS software packages, it may not be possible to integrate legacy COTS/GOTS applications into the ITA. These packages should be scheduled for replacement.

4.3.7.3 Tailored COTS/GOTS Applications

Some COTS/GOTS software packages can be tailored to a great degree to meet specific functional needs. Those products that make extensive use of APIs to interface with other applications and system services are generally the more flexible and robust. Others make use of proprietary interfaces to system services. As with custom applications, the HCFA Application

Architecture model supports the concepts of process wrappers, data wrappers, and data pumps when necessary to integrate purchased applications into the architecture.

4.3.8 Deployment Considerations

The determination of how to distribute an application is dependent upon and derived from business and user requirements. The approaches available for the distribution of application components are numerous and range from replicating applications on each network node to organizing applications on a functional basis and placing each functional grouping at a single location. Development teams must independently evaluate each application, the application data, and the user requirements to determine the best method for distributing application components and enterprise data. The appropriate model for application deployment and the correct number of discrete platforms to use are both business and systems-engineering decisions.

The paradigm for distributed computing has evolved from the traditional two-tier client/server model to an n-tier model of computing in which the components of an application are distributed across multiple nodes within a network. A strategy based on an n-tier distributed computing model has a number of benefits. For example, the application's scalability, performance, and reliability can be increased, and server layers can be replicated and distributed across any number of servers to boost system availability.

The n-tier approach also increases flexibility. Application layers can be modified quickly in response to changing business rules or economic conditions. Application service components may be placed anywhere in the infrastructure, so system administrators can easily reconfigure the system load. This n-tier model of computing can be depicted as a logical three-tier computing model, meaning that there is a logical, but not necessarily physical, separation of processes. The tiers then communicate through a set of widely accepted and industry-standard protocols, services, and software APIs.

The salient point here is that layered application designs provide inherently more flexibility for distribution and deployment in a way that optimizes use of the available computing infrastructure.

The n-tier model of computing, based on the tools, standards, and techniques precipitated by the Internet and associated market forces, represents an evolutionary technical alternative to traditional client/server applications. The Internet provides significant business value to HCFA, with its ready-made infrastructure for network connectivity and tool-enabled capabilities. Internet network and application protocol standards allow for the rapid sharing of information, dynamic application deployment, and leveraged network operations. Many architectural issues, such as platform independence and data access middleware, are already being addressed using this strategy. The Internet protocol suite consisting of TCP/IP (communication), FTP (file transfer), SMTP (mail), and HTTP (World Wide Web) has enabled the proliferation of portable Internet applications.

Internet tools and standards-based APIs provide the basis for HCFA's application development and evolution to the desired target architecture. However, the Internet is not a panacea. Exceptions to this strategy include high online-transaction requirements, advanced security requirements, and strict levels of guaranteed online performance.

4.4 Guidance

An architecture provides a clear pathway that guides the implementation of systems during the transition to an established set of values and principles.[7] It does not represent the existing infrastructure or application requirements, or the design or blueprint for any such system. Because business processes and IT are dynamic, we must have some idea of where we are going and how we will get there. Making effective use of both business processes and IT requires the use of guidelines and templates to establish and maintain consistency. Guidelines are needed to prevent inconsistent interaction between applications developed over time to support business processes—interaction not only between new applications, but also between new and legacy application systems as well.

HCFA's IT direction as set forth in the IT Vision, IT Objectives, and IT Guiding Principles described in Volume 1 guides us onto the pathway that leads toward the target ITA environment. The IT Objectives broadly describe where we are going, and the IT Guiding Principles describe how we will get there. Principles typically are not always well-understood, and assumptions can differ among staff. Because principles form the basis for determining the architecture, more-specific guidance is needed to further assist decision-makers involved in application development activities that help to advance HCFA from its legacy environment toward the target Application Architecture.

Two groups within HCFA form the primary target audience for the guidance presented in this section:

- Group 1 includes IT project leaders, system designers, application programmers, and decision-makers engaged in the development of business applications.
- Group 2 includes IT investment portfolio managers and decision-makers concerned with reviewing facts pertinent to IT investment proposals and deciding which proposals should receive funding approval.

In order for HCFA to evolve toward the target environment in terms of new and legacy application development, certain practices (standards) must be followed. These standards are to be considered compulsory, and are discussed as **design principles** in Section 4.4.1.

When reviewing IT investments for funding approval, Group 2 may use these principles for acceptance criteria to ensure that IT investments comply with the target architecture, as required by the Office of Management and Budget. The design principles also provide

[7] "From Mainframes to Distributed Computing: The Technical Issues," Gartner Group*, Strategic Analysis Report*, 24 July 1998.

standards by which Group 2 may evaluate compliance with the ITA during IT project compliance review processes (see Volume 7 – Management and Governance).

In addition to compulsory standards, Group 1 should observe other **recommended best practices** to the extent possible in application development projects. Doing so will advance HCFA further toward the target ITA environment. These highly desirable practices are discussed in Section 4.4.2. Group 2 may use the recommended best practices when making discretionary funding decisions between competing IT projects that meet compulsory standards. When funds are limited, Group 2 may give preference to IT projects that advance HCFA furthest toward the target IT environment.

4.4.1 Design Principles

For each design principle listed in this section, a brief explanation is provided along with pertinent implications for HCFA to consider. IT projects executed under the Application Architecture will be required to adhere to these design principles. The implications cited in this section will need to be addressed explicitly within project plans as they apply to particular development efforts.

The design principles listed in this section are a starting point and are not all-inclusive. As HCFA gains experience using this guidance, the list of principles may be modified. No priority is implied by the order in which the principles are listed.

1) Design Applications to Mimic a Business Process

The logical boundary of an application shall be discrete to the business process it supports and shall remain inviolate.

In doing so, the scope and complexity of each application is contained, thus avoiding the creation of monolithic systems.

Implications:

- Discrete business processes must be defined through a rigorous requirements-determination process.
- Emphasis on this approach must be reinforced throughout appropriate phases of the SDLC.
- Enforcing compliance with this approach may be difficult.
- Standards for mapping application logic to business processes must be established.
- Programmers and analysts may need additional training in order to understand how to implement applications under this approach.
- There may be a specialized role for requirements developers within HCFA.

2) Layer the Design of Applications

The design of applications shall be logically divided into three discrete layers: the user interface layer, the business logic layer, and the data access layer. A messaging interface and a data access middleware interface will provide the method for communication between the logical layers. The logical layers shall not be violated.

This approach produces a loosely coupled, adaptable application design.

Implications:

- An increased number of smaller application modules will need to be integrated and managed as a result of this approach.
- Training will be needed to improve staff skills with this design approach.
- There may be a specialized role for application designers and developers within HCFA.
- Rigorous design review throughout the SDLC is the primary enforcement vehicle.
- There may be a need for organizational realignments to support the layers.

3) Use Standardized Messaging between Layers

Service request and response mechanisms between application layers shall be standardized and message based.

In layered application design, a higher-level layer requests services from a lower-level layer in order for an application function to execute successfully. Modern techniques for having an application request a service and receive a response involve the use of messaging. The use of standardized messaging formats at each layer simplifies application development.

Implications:

- Reinforcement of this design approach must be incorporated throughout the SDLC.
- COTS products are available as standard technologies to implement this approach.
- Use of object-oriented technologies can facilitate implementation.

4) Design for Security Up Front

Applications shall be designed and developed to incorporate IT security policies at the beginning and throughout the SDLC.

Security policy cannot be implemented effectively as an afterthought.

Implications:

- Security service modules are primary candidates for standardized application components.
- Safeguards and a repository are necessary for shared or common code that implements security.
- Security requirements may drive how the design of an application is engineered.

5) Design Applications for Reuse

Applications shall be designed as modular, loosely coupled, and reusable components.

Reuse of application components simplifies application design, reduces development effort, and, when combined with a layered design approach, accelerates creation of adaptable applications.

Implications:

- A culture of software reuse must be established among application designers and programmers within HCFA.
- Hardware deployment decisions must be deferred until after application design to avoid imposing artificial design constraints.
- Reuse of common application components increases the failure risk (fault tolerance) of other applications should coding problems occur.
- Reuse of shared application services increases exposure to a single point of operational failure.
- Standards for appropriate levels of granularity for reusable modules must be established.
- HCFA must hire and retain skilled employees and invest in training for existing staff to strengthen programming skills in order to adopt a reuse culture.
- Emphasis on this approach must be reinforced throughout appropriate phases of the SDLC.
- Additional effort is needed to establish component libraries for administering reusable modules.
- Documentation of modules containing reusable functions must be made available (or its existence communicated) to other application designers throughout HCFA.
- There is a potential for a specialized role that would support application designers and developers within HCFA. Application designers possessing an overall view of the business process can work with the appropriate specialists during design review.

- Modules designed for reuse may slightly degrade application performance.
- Up-front costs to design and develop reusable modules may increase but would provide a return with each subsequent reuse.
- Appropriate incentives are needed that encourage software reuse among HCFA's contractors and operational environments.

6) Design Applications for Portability and Platform Independence

Applications shall be designed for portability and platform independence.

Portability is the ability to deploy (or redeploy) all or parts of a business application (i.e., user interface, business logic, data access) in the manner most optimal for exploiting available infrastructure resources. Platform independence means that business application logic is not dependent upon the specifics of a particular vendor's hardware or operating system platform.

Implications:

- Care must be taken to avoid design constraints that hold HCFA business applications to the lowest common denominator hardware platform.
- In designing and developing business applications, deployment decisions should be deferred until the implementation phase. Deployment engineers, not application developers, should be responsible for determining the best strategy for platform deployment.
- Use of proprietary operating system or hardware extensions must be avoided within the design of business application modules. However, it may be appropriate to use such extensions within the design of specific service modules.
- Future business application modules would need to be developed in languages that are platform independent (e.g., C, C++, Java).

HCFA has limited experience in languages that are platform independent. Most HCFA legacy systems are written in COBOL or M204 User Language, and both of these languages are IBM-MVS operating system dependent. Investments in training that will enable HCFA staff to program in C or C++ will be necessary; otherwise, future application development will need to be outsourced.

7) Use COTS/GOTS

COTS/GOTS products shall be used whenever possible. Custom-developed software may be used instead of commercial/government off-the-shelf products only when warranted and justified.

In cases where a business process is not unique and does not provide competitive advantage, COTS/GOTS products that satisfy HCFA's needs are preferable to custom-developed solutions. The implementation of COTS/GOTS products involves less cost and risk compared with custom design and development efforts.

Implications:

- COTS/GOTS products normally do not satisfy 100% of business requirements, so some customization is usually needed. The extent, difficulty, and expense of customization must be considered in arriving at make-versus-buy decisions.
- Criteria and thresholds for make-versus-buy decisions need to be determined in cases where requirements are not satisfied 100%. How do mandatory versus discretionary requirements influence the decision? Is an 80% satisfaction rate enough?
- The stability/continued viability of the vendor offering a COTS/GOTS solution needs to be included in the evaluation criteria for make-versus-buy decisions.
- Other pertinent factors in make-versus-buy decisions include HCFA's commitment to maintaining the infrastructure to support a given COTS/GOTS product, as well as roles and responsibilities for product support.
- The manner in which customization is accommodated within the COTS/GOTS product must be included in the evaluation criteria. Modifying the source code as a means of providing customization is prohibited. In some cases, changing the business process to better take advantage of inherent product capabilities is worth considering.

8) Promote the Use of Web-based Technology

Web-based technology shall be promoted for use in Agency applications.

The Internet and its associated Web-based technologies and standards have fundamentally and permanently changed the information systems landscape. These technologies have become a ubiquitous, economical, and standards-based resource for worldwide dissemination of and access to information. Web-based technology standards can be applied to transaction processing and information processing application designs. Web browsers have become the standard user interface for distributed business applications.

Implications:

- HCFA must strengthen the security of its network and database operations environment in order to exploit enterprise use of Web-based technology in application designs.
- A Web-based strategy needs to be articulated in the ITA for HCFA.
- HCFA must invest in training for its staff in order to enhance staff skills in designing Web-based business applications and in operating a production systems environment using Web-based technology.
- Database management, administration, and security standards and procedures must be assessed to incorporate the expanded use of Web-based technology.

9) Enable Automated, Active Information Delivery Technology

The automated, active delivery of information shall be enabled across the enterprise.

Information delivery applications are decision support tools that provide knowledge workers (analysts, researchers) and decision-makers with access to enterprise information. An automated, active information delivery model uses Web-based technology that enables information from databases to be automatically retrieved and delivered to a user's designated location (mailbox, folder, etc.) as soon as the information becomes available. This accelerates decision-making processes because users no longer need to manually request updated information each time that it becomes available.

Implications:

- The indiscriminate use of this technology may place heavy demands on the database and network infrastructure.
- A strategy must be articulated in the ITA to address appropriate uses of the active information delivery model in conjunction with a "publish and subscribe" model, in which predefined information reports are routinely published and disseminated to users based upon subscriber lists.

- HCFA has no practical experience with this technology. The Agency must invest in contractor resources and staff training to gain the expertise necessary for effectively implementing this model of information delivery.
- Not all applications are candidates for using this information delivery model. Explicit criteria for its use must be defined.

10) Separate OLTP from OLAP

Online transaction processing (OLTP) applications shall be separated from online analytical processing (OLAP) applications at deployment.

Performance demands are vastly different for these two classes of applications. Great difficulty in optimizing the use of resources is incurred when these types of applications are deployed on a shared operating platform.

Implications:

- Additional platform infrastructure may be needed in order to separate the workload.
- Distinctions between the business application requirements of transactional data processing and those of information retrieval processing need to be understood and defined during the design phase.
- Business applications providing access to HCFA transaction data or decision support information should perform these operations transparently. Users should be insulated from platform infrastructure considerations.

11) Design OLAP to Use Data Warehousing

Data warehouse concepts shall be leveraged to reduce the burden of development and to accelerate decision-making.

Data warehousing has matured in methodology such that IT solutions can be acquired and deployed using COTS products and little custom programming. Doing so reduces HCFA's level of effort and accelerates the provision of IT solutions for decision support.

Implications:

- An enterprise-wide data warehouse methodology and strategy must be defined for HCFA.
- Information subject areas contained within HCFA's data warehouse, as well as access methods, must be specified, communicated, and made available enterprise-wide.
- Methodologies, disciplines, tools, and techniques for implementing data warehousing differ from those used for transaction processing applications. Specialized skills, roles, and training for HCFA staff will be needed.

- Information engineering and data administration are essential to success.

12) Design for N-Tier Client/Server

Applications shall be implemented using an n-tier client/server model.

Adaptable applications have a modular, layered, and loosely coupled design and are deployed onto a distributed computing infrastructure. Maximum adaptability is achieved when implementing an n-tier client/server model.

Implications:

- Knowledge and expertise in designing and implementing n-tier client/server applications must be acquired to assist HCFA.
- Technology for effectively deploying and managing n-tier client/server systems within HCFA's' distributed computing environment is needed.
- Pertinent development and deployment standards must be identified. Design templates and software module templates would facilitate the establishment of standards across the enterprise.
- Additional investments in client/server technology infrastructure may be needed.
- More time may be needed for technical staff to adjust to the change from conventional design approaches. Proposed project schedules should account for this potentiality.
- Technical design and development reviews should include assessment criteria for adherence to this approach.
- Not all projects may need to adopt this approach. Appropriate criteria should be established to determine when this approach is warranted.
- Significant effort will be required to plan and carry out the transition or reengineering of legacy systems under this concept.

13) Use Standard Methods

Standard application development methods shall be adopted.

Standard methods for application design and development help to ensure interoperability and access to enterprise information by business applications throughout HCFA.

Implications:

- Different types of business needs and different types of applications developed to access information warrant different standards for application development. More than one application development standard (methodology) is needed within HCFA.
- Appropriate methodologies must be identified for application development within HCFA. The current standard (HCFA Information Systems Development Guide) must be made current, as it is no longer viable for serving this purpose.
- Training will be necessary in order to prepare staff to accept an improved methodology, particularly if more than one is adopted.
- Rigor in enforcing any new methodology must be tempered by the need for HCFA to gain experience through its continual use.

14) Use Prototypes and Pilots

Prototypes and pilots shall be used to achieve a working system first.

Prototypes and pilots are alternative means of exploring the appropriateness and feasibility of employing technology within HCFA while minimizing the level of effort, cost, and risk associated with deployment. Prototypes are proof-of-concept IT projects of limited scale and scope that are used to assess the technical feasibility of a custom-developed business application solution. Pilots are limited scale, initial deployments of IT solutions and are designed to prove the viability of an application for a specific business purpose. Prototypes are typically unnecessary for COTS business applications. Pilots may employ all custom-developed applications, all COTS applications, or combinations of both. Rapid Application Development (RAD) methods and tools facilitate the design and development of prototypes.

Implications:

- Prototypes have a way of becoming the “production” implementation once they are proven to satisfy a given need. Considering that adaptability is an essential concept to be achieved through the Application Architecture, a formal prototyping process should be incorporated into the HCFA SDLC (see Section 4.6.1.2).

4.4.2 Recommended Best Practices

For each recommended best practice listed in this section, a brief explanation is provided along with pertinent implications for HCFA. HCFA cannot require each application development project to follow these recommended best practices, although their use is highly encouraged. The implications should be appropriately weighted for each IT project under review or pending approval.

The best practices recommended in this section are based upon analyses and recommendations from the Meta Group, Inc., Gartner Group, Inc., GigaGroup, Inc., and leading IT industry consulting services organizations. These recommendations may be modified, deleted, or supplemented as HCFA gains experience through using this guidance. No priority is implied by the order in which the best practices are listed.

1) Object-Oriented Design

Application development and delivery should evolve toward an object-oriented approach.

Object-oriented design and object-oriented development differ from traditional approaches because they are based upon precepts established for modular, standardized application components and software reuse. Modern business applications and information systems are being designed using object-oriented techniques.

Implications:

- Tools and technology standards that support object-oriented design and development differ from traditional tools and techniques.
- HCFA staff will need training to become skilled at object-oriented design and development.
- Object-oriented design and development tools and methods may impose infrastructure requirements that will need to be addressed.

2) Business Event-Driven Design

The logical design of applications should be driven by business events.

A fundamental premise behind this notion is that business "events" and the business processes they trigger are interactive, rather than "batch" occurrences. Business processes are optimized when the applications that support them are promptly executed, as opposed to transactions being held for batch execution at some later point in time. A typical HCFA business event could be a beneficiary electing to enroll in a Managed Care Plan from traditional fee-for-service insurance coverage. The business processes that this might trigger are predetermined by Medicare business rules, which dictate the sequence of actions to be taken by HCFA, its agents, and business partners in response to a specific event. Fundamentally, this concept would result in all HCFA business applications being designed for interactive — rather than batch — execution.

Implications:

- Further analysis of each HCFA business function is necessary to decompose them into discrete business processes in order to identify trigger events, dependencies, and interactions with other processes.

- Opportunities for business process reengineering (BPR) may be identified as a by-product of the analysis mentioned above.
- HCFA's business is not changed as a result of this design approach, but it does represent a significant culture change in the way in which business and system requirements are defined.
- Use of object-oriented analysis and design techniques facilitate this approach.
- Applications must be designed using a client/server approach.

3) Programmer Specialization

Specialized roles for programmers should be defined.

The increased sophistication of advances in today's technology is challenging programmers to maintain proficiency in all aspects of systems design and development. Programmers must specialize in technology used for designing and developing business applications in order for HCFA to leverage investments in IT development resources across the enterprise. Centers of excellence based upon specialized roles defined for programmers in the design and development of business applications can enhance IT solution delivery throughout the enterprise. Examples of specialized roles include requirements specialists, user interface specialists, common module and reuse specialists, messaging specialists, and data access middleware specialists. By redefining application delivery processes (from a stovepipe approach to a model based upon centers of excellence), HCFA can maximize the benefits of investing in contractor resources for application development and training application programming staff.

Implications:

- Explicit roles for programmer specialization, as well as changes to application delivery processes, need to be defined in context with other roles in the enterprise, along with changes to HCFA's SDLC methodology.
- Significant changes in Federal staff job descriptions may give rise to considerations of the employees union.
- HCFA must make a commitment to training appropriate for the roles that are defined. The approach will help focus how training dollars are spent.

4.5 Competencies

People and the IT skills they possess are essential to HCFA's ability to evolve to the target Application Architecture environment. The complexity of distributed computing as compared with legacy mainframe systems, and the scarcity of skilled IT resources, will make it difficult for HCFA to retain sufficient permanent staff with the requisite competencies. Consequently, HCFA will continue to be reliant upon a blend of permanent staff and contractor resources with the requisite skills working in a collaborative environment to achieve application delivery.

Improving application delivery to the enterprise in an environment of increasingly fewer permanent IT staff and increasing shortages of skilled contractor IT resources is an important consideration for HCFA. Application development skills needed by HCFA must be clearly understood so that responsibilities can be appropriately allocated between central IT support functions and the business units. HCFA must implement and institutionalize a continuous review process to identify the skills that are critical to application delivery processes under the target Application Architecture approach. Once these requirements are determined, HCFA will be able to better focus its permanent staff skills and training needs, and to determine its contracting requirements. Resources can thus be deployed and shared effectively between the central IT functions and the business units.

4.5.1 Specialized Skills

Several specialized skills are important to the successful delivery of business applications under the target Application Architecture approach. The specialized skills discussed in this section apply to both transaction processing and decision support applications, whether custom developed, COTS, or an integrated combination of both.

4.5.1.1 Requirements Specialists

Business functions are collections of related business processes. Design guidance for the target environment calls for a move toward modular, granular applications that mimic a discrete business process, rather than monolithic systems supporting multiple processes. Methods for building applications under this approach must be redefined, starting at the requirements analysis phase of system development, to ensure that each discrete business process is uniquely identified, along with its relationships and dependencies to other business processes.

Requirements analysis efforts in the future must ensure that each business process is discretely defined and well-documented, and that data access needs are well-understood. This includes being aware of available databases and new data requirements. Specialized skills in business process analysis, systems analysis, and data management analysis are needed to accurately determine business application requirements under the target approach.

4.5.1.2 User Interface Specialists

User interfaces to business applications have evolved from traditional character-based presentation to graphical (Windows-based) presentation as the standard. In addition, the emergence of Call Centers to support centralized customer-service operations has given rise to alternative devices, such as computer telephony interface (CTI) and automated response units (ARUs), that access business applications. The fast evolution of Web-based technology has the potential to extend user interface alternatives even further as browsers are given more robust sound and video capabilities.

With few exceptions, most business application logic should be independent of the way it is presented to the user — character-based versus graphical screens. However, traditional approaches to designing HCFA legacy systems fail to leverage this fact, resulting in redundant

applications performing similar functions for the same business process. For example, consider a typical HCFA scenario of two customer-service representatives responding to requests from separate beneficiaries to change personal data in the Enrollment Database (EDB). One uses a character-based (3270) device to access an online application that updates the EDB. The other uses a graphical user interface (GUI) device to access a separate application that also updates the EDB. Both applications must enforce the same validation and consistency logic for input data in order to update the EDB successfully. Since the applications were designed for different operating platforms, using different languages and user interface methods, the logic for validating data input is redundantly coded in separate online application programs. This increases system resource demands and maintenance costs.

In the target design approach, data input validation logic would be coded once, in a language that is portable to different operating platforms, and reused as a common application module that is accessible to any user interface device. This approach promotes reusability and ensures that regardless of the user interface device, the same business application logic would be executed, thereby reducing system maintenance costs. Designing business applications in this manner requires employing specialized skills and expertise in the different user interface devices found at HCFA today, staying abreast of advances in technology, and leveraging this expertise across the enterprise.

4.5.1.3 Software Reuse Specialists

Some business functions within HCFA perform similar processes and need access to like data. An application that supports a business process that is common within HCFA presents the opportunity for software reuse. Software reuse typically occurs either by embedding common code segments that perform a specific function into different application modules during program compilation, or by calling separate executable modules to perform a specific function at run time.

Regardless of the technique employed, application developers must make a concerted effort to leverage software reuse. Procedures for identifying, cataloging, and publicizing common code segments and callable modules throughout the enterprise must be established.

Software reuse must be promoted throughout HCFA in order for the benefits to be realized. The Agency must create a software reuse culture and commit the resources necessary for it to flourish. HCFA must identify the specialized skills and expertise required to design and implement a comprehensive software reuse program. Reuse specialists will work with other specialists to design and develop methods that can be implemented enterprise-wide, thereby reducing the Agency's overall application development burden.

4.5.1.4 Database Access Specialists

Database access logic must be separated from business process logic in the design of future business applications in order to be consistent with the target Application Architecture. Isolating data access from business logic facilitates the creation of adaptable applications, helps to

contain uncontrolled data proliferation problems, and permits comprehensive approaches to data management.

Application programmers cannot effectively serve their primary role of delivering business applications to their customers and at the same time deal with systemic problems of data management. Preferably, application programmers should rely upon database managers and data access technicians to satisfy needs for access to enterprise data. Data access software and services are primary candidates for a software reuse program within HCFA.

Effective use of DBMS technology to enhance data and information access throughout the enterprise requires a commitment to retain competent staff (or use contractors) and keep their skills current. As the volume of data that HCFA manages continues to increase, and demands for information follow suit, skills and expertise in DBMS technology will become even more critical.

4.5.1.5 Messaging Services Specialists

Implementing loosely coupled, adaptable applications requires a design approach that is message-based. Messaging provides the essential glue that allows functional components of business applications (modules) to communicate with one another and request services of the platform infrastructure, such as data access or security.

HCFA has limited experience with messaging technology. Only a few products are widely used in the industry, and a variety of messaging techniques using these products can be implemented. However, in order for HCFA to successfully leverage messaging as an enterprise strategy, a consistent approach must be defined based upon standards.

As with most new technology, how to best implement messaging in HCFA's environment must be learned through experience. HCFA must invest in acquiring and upgrading the skills of application programmers experienced in the use of messaging technology. Outside expertise would be necessary to assist HCFA initially, and methods would need to be developed in order to leverage the use of messaging across the enterprise. Specialized skills in implementing messaging technology could be an enterprise resource used to support multiple application development projects.

4.5.1.6 System Services Specialists

System services cover a broad spectrum of support necessary for implementing and operating business applications in a distributed computing environment (of desktop, mid-range, and mainframe processors). These services support systems interoperability, transaction management and workflow, naming, system calendars and timing, error handling, and other operational controls.

Ideally, application programmers should be able to focus on solving the business problem rather than dealing with the complexities of hardware platforms. Unfortunately, the complexities of a

distributed computing environment can consume a disproportionate amount of the time estimated for completing software development projects.

System services are an element of application design that, in the current environment, tends to be hardware platform and operating system specific. There are gaps in standards across the spectrum of services needed to implement business applications in a distributed computing environment. HCFA will need to develop a nucleus of skilled technical specialists in order to provide a broad range of system services without resorting to proprietary implementations.

The technique used to implement system services will influence the skills and resources needed to support application development. By necessity, system services specialists must work closely with other specialists (i.e., user interface, data access, and security specialists) to design and develop enterprise implementation methods.

4.5.1.7 Security Services Specialists

Implementing security for business applications in a centralized mainframe environment can be a relatively straightforward endeavor because few variables exist and security technology in the mainframe operating environment is very mature. By contrast, providing security for business applications in a distributed computing environment is a formidable challenge for HCFA. Security technology in this environment lacks the stability provided in the mainframe environment.

Providing security for distributed application systems is a technically complex undertaking. The implementation of security must be comprehensive and thorough in order to avoid exposure, and this makes it a high-risk endeavor. Security must be integrated seamlessly across hardware, software, network, and data components to minimize exposure to threats. For HCFA's large number of geographically dispersed enterprise users, the complexity and risk factors increase considerably.

Technical alternatives for security within a distributed computing enterprise are relatively few, and industry standards are still evolving. HCFA has little experience implementing enterprise security off of the mainframe platform, and the target security architecture is yet to be defined. Once the Security Architecture has been defined, the technology standards, products, and skills required to implement enterprise-wide security can be more clearly identified.

The technical details of HCFA's security implementation must be limited to only those with a legitimate need to know. Therefore, HCFA must limit the number of staff possessing the detailed knowledge necessary to implement security, and this limited staff must possess the specialized skills needed to provide technical support to application developers throughout the enterprise.

4.5.1.8 System Management Services Specialists

System management services primarily cover the functions of software distribution, software installation, software rollback, application program startup and shutdown, and application

program alerts and alarms. Many of these services must be designed within business application modules and integrated with other services (such as messaging, security, and system services).

A variety of COTS products are available for implementing the desired features. These products usually provide APIs for implementing optional features, but considerable expertise is required that is typically not found among business application programmers. The complexity of designing and engineering effective solutions to support an enterprise approach to system management should not be underestimated.

Implementing system management services can be simplified with proper expertise and the right approach. Specialized skills and expertise are necessary to develop methods that can be leveraged by the enterprise to ease the application developer's burden.

4.5.2 Centers of Excellence

Rapid advances in IT will continue to challenge the ability of programmers to maintain proficiency in all aspects and disciplines of application development. When faced with complex design, development, or implementation issues, application programmers should know where and from whom to seek expert advice and assistance within HCFA.

Centers of excellence are a means of dedicating IT resources to given areas of expertise. These dedicated resources can assist in identifying best practices and in building standard methods for leveraging their specialized expertise throughout the enterprise. Centers of excellence can provide expert advice and assistance to application development projects throughout HCFA that require a short-term, focused skill. They can also assist application development projects in complying with ITA standards.

The specialized skills discussed in the previous section are candidates for centers of excellence. Centralized IT support functions within HCFA (OIS and OICS in particular) already serve as competency centers for some IT expertise, primarily in database and technology infrastructure.

Much of HCFA's application development expertise and resources come from project teams within the business units. These project teams will be the leaders in developing specialized skills to implement business applications in HCFA's distributed computing environment. The development of strategies for leveraging specialized skills and expertise within the business units to benefit the enterprise represents a resource management challenge for HCFA.

Establishing centers of excellence does not imply creating new organizations, nor does it require collocating or centralizing staff. However, someone must manage the responsibility for identifying the expertise that is available within existing organizations and coordinate with project managers and business units to optimize the sharing of resources. The roles and responsibilities associated with the centers of excellence will be determined in the course of refining the Management and Governance process.

4.5.3 Organizational Considerations

There are a number of organizational considerations for HCFA regarding the development of specialized skills and the creation of centers of excellence to support the enterprise. A role-based application delivery model may prove more effective than HCFA's current stovepipe project execution model. Our stovepipe execution model holds project teams within the business units fully responsible for every aspect of business application delivery. HCFA's central IT functions provide support primarily related to infrastructure and database technology.

In a role-based application delivery model, project teams would share development responsibility and resources with central IT functions and use centers of excellence for specialized expertise. Each specialty area would have defined roles in fulfilling application delivery for the enterprise using standard methods that are repeatable. Application development support responsibilities would cycle through different organizations within and outside of the initial project team, depending upon where the needed expertise resides.

Project managers would still oversee the entire project, ensuring that the delivered business application satisfies customer needs. Infrastructure and center of excellence managers would need to work closely with project managers in order to manage priorities and coordinate resources under this approach. Few incentives exist for central IT managers to share resources and risks with business unit and project managers. This kind of model represents a cultural change for HCFA and will be implemented over time in conjunction with the Management and Governance process.

4.6 Standards

Implementing a business application includes the definition, design, development, testing, installation, and functional operation of a system that supports a business process as defined by the business owner. Depending upon the type of application needed to support the business process, each type may have different methods of implementation and associated standards. This section describes appropriate standards for developing business applications in a manner consistent with the Application Architecture design principles.

Standards fall into two main categories: methodology and tools. In this discussion, methodology (discussed in Section 4.6.1) represents HCFA's SDLC methodology for application development. Tool standards (discussed in Section 4.6.2) identify the specifications established by industry consensus organizations and vendors supplying software applications and commercial products for use in application design, development, and deployment.

4.6.1 Methodology

Standards for application development are based upon an SDLC methodology. An SDLC methodology outlines the high-level phases of the application development (AD) life cycle, describes the detailed procedures to be executed within each phase, and identifies the inputs and outputs of each phase. Methodologies for application development and management are in their second and third generations, reflecting extensive experience with the use of tools. An

effective tool infrastructure will enable HCFA to more effectively gather requirements and to plan and estimate the AD effort so that it is a more predictable and productive experience.

The methodologies and tools used for application design and development can differ among the types of business applications. For example, a methodology used for developing a transaction processing application would be quite different from that used for a decision support (or data warehouse) application.

Significant gaps exist between HCFA's legacy SDLC methodology and the needs of the Application Architecture. The following sections provide brief overviews of HCFA's current legacy SDLC methodology and the revised methodology framework that is necessary to support the target Application Architecture.

4.6.1.1 Legacy SDLC Methodology

HCFA's current SDLC methodology is described in the HCFA Information Systems Development Guide (HISDG). The HISDG presents a largely structured approach that is based upon IS philosophies prevalent in the era when most HCFA legacy systems were developed (the 1970s and 1980s). It has not been formally revised since 1994. Industry standards, application development tools and techniques, and HCFA's own technology infrastructure have undergone radical changes since that time.

The methodology presented in the HISDG is unacceptable for current and future application development needs. The major deficiencies include the following:

- Gaps exist among tools that automate application development processes across life cycle phases; therefore, manual steps are required to bridge technology gaps between phases. The manual steps only increase the application development burden, with little payback to the project.
- Robust support for rapid application development methods and tools is lacking.
- Guidance to enhance the delivery of projects of differing scope and complexity is lacking.

These deficiencies must be addressed in order for an enterprise methodology to be effective. Recognizing this, HCFA has engaged in a reassessment of the HISDG with an eye toward enhancement. Enhancing the methodology will ensure that modern application design and development methods, tools, and techniques are used to facilitate the construction of business applications in a manner consistent with the Application Architecture approach.

Factors to be considered in enhancing the methodology include the following:

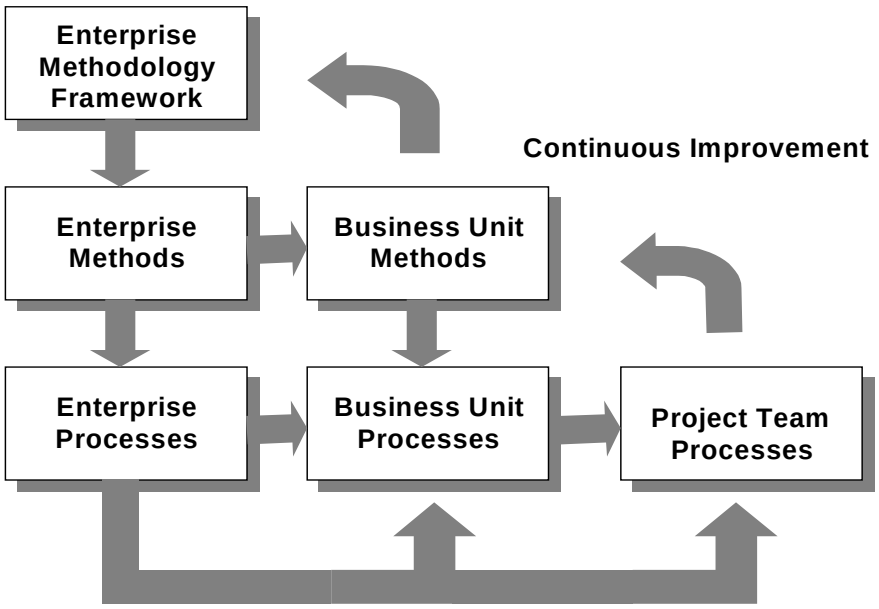
- HCFA will continue to be dependent upon maintaining its legacy system operations while working toward reengineering or replacing them based upon changed business needs. The enhanced revised methodology must support a hybrid environment of legacy system maintenance and new business application development activities.

- The IT staff within HCFA's business units must respond with solutions to new business needs more timely than in the past. The enhanced methodology must accommodate traditional software development methods as well as rapid development approaches.
- Responsibility for application development is dispersed among business units throughout HCFA, requiring collaboration among a variety of stakeholders in providing IT solutions to the enterprise. The enhanced methodology must support individuals as well as workgroups in multiple locations who are collaborating on application development projects across the enterprise.
- The enhanced methodology must accommodate projects that differ in size, scope, and complexity, and must require that a minimum number of phases and steps be performed to successfully accomplish the effort.
- Both transactional and decision support applications are needed by the business units for access to enterprise data. The enhanced methodology must provide separate procedural standards and tool sets appropriate for both types of needs.
- Centralized IT functions have limited responsibility for business application development within HCFA. These functions, for the most part, provide infrastructure operation, database administration, and data management support. The enhanced methodology must include a new centralized IT role that involves administering the sharing and reuse of application modules and service components across the enterprise.
- HCFA intends to acquire a comprehensive, integrated AD tool set that automates the SDLC phases to the extent possible. The enhanced methodology must incorporate the use of an integrated tool set for management and control of key SDLC processes in an effective manner.
- Establishing a culture of reuse is a primary tenet of the Application Architecture. The enhanced methodology must enable and promote maximum sharing and reuse of AD resources (requirements documentation, data definitions, component modules, etc.) across the enterprise.
- HCFA must adopt methods that enable the Agency to attain (no less than) Level 2 compliance with the Software Engineering Institute's Capability Maturity Model (SEI CMM). HCFA aims to use the SEI CMM at a future point to measure the maturity of our application development and management processes, and to evaluate those processes for continual improvement. The enhanced methodology must advance HCFA toward this goal.

4.6.1.2 Revised Methodology Framework

A revised methodology framework that supports the Application Architecture serves as the foundation for leveraging, applying, and continuously improving application development methods, processes, and techniques throughout HCFA. Exhibit 4-18 depicts this concept.[8]

EXHIBIT 4-18. HCFA ENTERPRISE AD FRAMEWORK



The vertical dimension of the approach identifies the levels of detail to be defined within an AD methodology. These levels consist of the enterprise *framework*, *methods*, and *processes*.

- The *framework* level defines the highest context for all application development disciplines or specialties within HCFA. This level describes the generic activities and deliverables addressed by the enterprise AD methodology (i.e., the high-level work elements and work products supported by the methods and processes within the methodology). Two primary AD frameworks must be supported by HCFA's enterprise methodology: a transaction processing AD framework and a decision support AD framework. A primary purpose of the framework level is to provide a common structure that facilitates the sharing and integration of methodology phases at the *methods* and *processes* levels among multiple business units and teams during project execution. The common terms and views established through use of the framework will enhance

[8] This enterprise AD approach is based upon the EDS Methodology Architecture found in the EDS Technology Policy, 1997, Electronic Data Systems (EDS).

consistency and coordination among the methods and processes within a methodology, and will facilitate the selection of appropriate methods and process activities to support a specific project. The framework will also allow HCFA to identify common links between different methodologies.

- The *method* level provides a collection of activities that describe individual aspects of the work to be performed. It also describes the general links or dependencies among these activities for planning purposes. Methodologies may differ, depending upon the type of enterprise framework (i.e., transaction processing or decision support). Each methodology may have methods that individually define a general approach to performing all or part of the work supported by the methodology. The individual methods do this by specifying the description, purpose, and relationship of specific work elements (tasks) and work products (inputs and outputs). The methods define the detailed links with other disciplines and interdisciplinary functions. In addition, the methods may reference specific techniques, tool standards, and other performance support for accomplishing a task or producing a work product.
- The *process* level applies the activities defined in one or more methods and reuses portions of other processes to best support the actual work to be performed. The process assists the selection and ordering of work elements as needed to accomplish a specific project, such as custom-developing new business applications, maintaining legacy systems, or acquiring and integrating packaged applications in a specific environment. Factors influencing the environment include operating constraints, interface requirements, resources, skills, and choices of tools. The work products necessary to support links to other process disciplines and interdisciplinary functions are integrated in sufficient detail to provide repeatability across multiple business units and teams throughout the enterprise. Associated tools are selected to reflect enterprise management and technology policy and standards.

The steps of a process may vary depending on the type of work the process supports and on the established policy for using the process. For example, one process may contain detailed procedures describing how to complete each step and work product, while another process may contain little more than identification, a brief description, specific ordering of steps to be accomplished, and the work products to be delivered.

Detailed, repeatable procedures, beyond those that focus on process control, can be inefficient and ineffective for systems engineering disciplines because they are decision-intensive and must be applied to various situations. Such disciplines are better served by higher-level processes that identify and order only the most common set of work elements to be performed and the work products to be delivered (the “what to do”). The higher-level work elements and work products, in turn, can point to relevant method activities that provide more detailed information on one or more ways to perform those work elements and to produce the work products (the “how to do it”). This approach balances the need for enterprise process maturity and consistency with local process flexibility to address unique business needs.

The horizontal dimension of the approach identifies the degree of customization available within a methodology. This dimension is divided into *enterprise*, *business unit*, and *project team* levels.

- The *enterprise* level defines the framework and its associated methods and processes that can be used across the HCFA enterprise. These enterprise assets provide a rigorous foundation for building methods and processes as well as project team processes.
- The *business unit* level defines methods and processes that can be used within HCFA's organizational business units. Business units use the enterprise framework, methods, and processes as templates that can be customized to reflect each business unit's specific needs. These needs include specific customer requirements, industry standards, or regulatory specifications. Business units also develop their own methods and processes to meet specific needs, such as niche products or services. Some examples are products for actuarial research and analysis, or services for technology infrastructure operations and management. A business unit's methods and processes are maintained and used repeatedly from project team to project team. Multiple levels of business units can exist between the enterprise and project team levels.
- The *project team* level defines processes that apply the methodology to actual work requirements, with an emphasis on the specific organizational and environmental conditions under which the project team performs. Project teams use methods and processes provided at the enterprise and business unit levels as templates that can be customized to reflect the project team's specific needs. These needs include additional customer requirements, project team standards, resources, expertise, and training. By following the project team's process, the project team produces a result that is predictable in substance, quality, and schedule.

The enterprise methodology framework described above uses three ingredients in a formula for leveraging HCFA's collective knowledge and experience. First, multiple levels of detail are recognized in the approach to, and definition of, enterprise methodologies. HCFA's enterprise methodology framework will likely contain multiple methods, and a method can be applied to multiple processes. This approach permits leveraging information at the highest level to which it applies and adds detail and specialization at lower levels as appropriate. As processes are found to use similar procedures, these procedures are built into the method level to be leveraged across multiple processes. As multiple methods are found to support similar AD activities, these activities are defined at the framework level.

Second, responsibility for defining processes is shared between central IS and business unit organizations across HCFA. This sharing is a cycle of development, deployment, and continuous improvement (see Exhibit 4-18). Following deployment, organizations can benefit from lessons learned to improve their processes. Business units and project teams can then provide feedback directly to the process owners to leverage best practices and improve design guidance.

Third, an innovative technique or effective procedure defined by one process owner can be made available for others to use and integrate into their own processes, thereby avoiding duplication of effort. This fosters a culture of reuse. Reuse minimizes the effort and cost expended to define methods and processes, and extends the value of these efforts by enabling project teams to share their work. A framework provides the structure for facilitating methodology phase integration and interchange. Methods and processes can be designed to promote modularity and reuse. This focus on reusing rather than redefining common activities allows method and process developers to take advantage of prior learning to continuously improve processes.

Industry has shown that process improvement reduces errors, increases productivity, and provides consistent results, thereby increasing end-user satisfaction. The enterprise AD methodology framework for supporting the Application Architecture fosters an environment that is process-driven, enterprise-supported, locally prescribed, and continuously improved. This framework will guide HCFA in its efforts to revise or replace the legacy methodology.

4.6.2 Tools

Application development tool standards are based upon specifications established by industry organizations and vendors that supply commercial software applications and provide tools for use in application design, development, and deployment. These standards assist HCFA by:

- Ensuring interoperability between application components;
- Ensuring compatibility of interfaces with other application software products;
- Ensuring interoperability with other infrastructure technology within the Application Architecture framework;
- Leveraging the commercial market viability of technology products; and,
- Providing for the orderly acquisition and evolution of AD products that include new features advantageous to HCFA, and eliminating those that are no longer useful.

HCFA needs well-established AD environments with capabilities that facilitate the selection of standard tools according to the criteria for a given project. These capabilities must also provide the enterprise with a framework for planning an entire IT services delivery effort when proposing new projects, as well as when actively engaged in the delivery itself. Many of today's tools offer substantial productivity increases; however, if an environment is too complex, the productivity gained in one area will be lost in another. The goal is to design and develop business applications using products and tools that minimize environment complexity and maximize productivity in supporting and maintaining the applications.

4.6.2.1 Tool Evaluation Criteria

The criteria for evaluating and selecting enterprise AD tools have been divided into five categories to facilitate further analysis of tool capabilities. The five categories are listed below:[9]

- **Business Issues** – Criteria that extend beyond purely technical considerations but are critical to the selection of an AD tool. Such criteria include staff expertise, resource assessments and training needs, product installation and ramp-up time, and costs and budget constraints.
- **Architecture** – Criteria for the underlying structure, adaptability, and longevity of business applications that can be created by AD tools. Technology infrastructure standards and business constraints, as well as the target Application Architecture approach, determine these criteria.
- **Engineering** – Criteria that address the planning, analysis, and design phases of the SDLC, which provide for the conceptualization of business applications. These criteria must consider outcomes from HCFA's methodology enhancement efforts.
- **Development** – Criteria regarding capabilities for generating functional application modules with maximum productivity and quality. These criteria must also consider the outcomes from HCFA's methodology enhancement efforts.
- **Operations** – Criteria pertinent to the execution of generated applications in a production operation environment. These criteria must take into account the outcomes of HCFA's enterprise systems management assessment and future strategy.

4.6.2.2 Tool Categories

Application development tools generally fit into one of several categories that are recognized throughout the industry and which quickly identify a tool's primary capabilities. Using these categories is beneficial in narrowing the focus of the AD tool evaluation effort. The categories are:

- **Integrated Computer-Aided Systems Engineering (I-CASE)** – I-CASE tools allow software engineers to automate much of the SDLC, particularly in the Architecture, Engineering, and Development categories. I-CASE tools are sometimes referred to as lower-CASE tools.
- **Upper-CASE** – Upper-CASE tools are computer-aided software engineering products that support planning, definition, analysis, and design activities of the SDLC.
- **Object-Oriented Analysis and Design (OOA/D)** – OOA/D tools support the analysis of business requirements and the design of business applications using object-oriented techniques.

[9] This approach is based upon the Comprehensive Tool Infrastructure (CTI) Framework, EDS Technology Policy, 1997.

- **Visual Programming Environments (VPEs)** – VPEs provide graphical utilities for designing some aspects of an application, including the GUI, menu, icons, and bitmapped images. This category includes most fourth-generation languages (4GLs) and lower-CASE tools with a visual interface.
- **Third-Generation Languages (3GLs)** – 3GLs consist of compilers, class libraries, and API tools that provide graphical and design features to specific languages.
- **Enterprise CASE (E-CASE)** – E-CASE tools are an emerging category of technologies that are expected to most comprehensively address application implementation requirements in the future.

Arraying the evaluation criteria against the categories provides a comprehensive enterprise AD tool selection framework to support further analysis by HCFA. Exhibit 4-19 depicts this framework. The "X" indicates the capability to satisfy the criteria for that category. The Business Issues criteria are non-technical and therefore do not appear in the matrix. These issues are covered separately in Section 4.6.2.3.

EXHIBIT 4-19. HCFA ENTERPRISE AD TOOL SELECTION FRAMEWORK

Criteria	I-CASE	OOA/D	VPE	3GL	E-CASE
Architecture – Application Topologies	X		X	X	X
Architecture – Environments	X		X	X	X
Architecture – Standards Compliance	X		X	X	X
Engineering – Planning	X	X			X
Engineering – Analysis	X	X			X
Engineering – Design	X	X			X
Development – Reuse	X		X	X	X
Development – Construction	X		X	X	X
Development – Integration	X		X		X
Operations – General Services					X
Operations – SW Configuration Mgmt					X
Operations – SW Distribution					X

4.6.2.3 Business Issues

Before AD tools can be evaluated, HCFA's business applications needs must be understood. Many variables need to be considered, including some of a non-technical nature. Not all business applications require the use of expensive, complicated tools, but neither can all applications be built using inexpensive tools, which typically do not scale well to support the growing business needs of the enterprise. Variables to consider include:

- The life span of the business application to be developed;
- The scale of use for the application;
- The degree of integration with legacy or future business applications;
- Availability of development resources and budget for the project;
- Development of staff skill sets;
- Time-to-market; and
- Organizational culture (i.e., business value and expectations of technology).

4.6.2.4 Architecture

The target Application Architecture provides a conceptual structure for the interoperability, adaptability, and longevity of custom-developed business applications. These benefits are facilitated by the use of AD tools that support industry standards, and multiple environments facilitate these benefits by aiding the creation of loosely coupled, n-tiered applications.

I-CASE and VPE tools usually support many of the target Application Architecture considerations, but the available architectural options are often limited. It is imperative that HCFA understand these limitations and commit to embracing the selected tool as an enterprise standard. The selected product must also conform to open systems standards in order for HCFA to avoid having its infrastructure become dependent on a single vendor.

Establishing an application architecture using 3GLs tends to offer little support and require more effort than I-CASE or VPE tools. 3GLs typically allow more flexibility in application development and more options for external interfaces and portability. However, the development and maintenance productivity of 3GL-based applications is lower than that of I-CASE and VPE applications.

To be consistent with the target Application Architecture, HCFA application developers need a tool that allows the creation of portable applications. This feature should ensure that the application looks native to (is designed to fit with) each user interface environment to which it is deployed. Strategies for maintaining portability across multiple database management systems, operating systems, and interprocess (process-to-process) communications mechanisms are

needed. The target Application Architecture approach isolates business applications from their underlying physical environment to increase portability and maintainability.

Support for layering and partitioning the business application should be incorporated into the generation capabilities of the AD tool. Dynamic repartitioning of an application allows processes to be moved or distributed at runtime to other operating platforms for load balancing or fail-over operations. Layering allows for the logical grouping and placement of application module components onto the appropriate platforms (i.e., the user interface onto the desktop, database access onto database servers, application services onto dedicated servers, etc).

Support for interprocess messaging, distributed objects, and database gateway middleware should be incorporated into the capabilities of the AD tool. These features implement standards such as Common Object Request Broker Architecture (CORBA) and Distributed Component Object Model (DCOM) technologies, which currently dominate the industry.

4.6.2.5 Engineering

Real business value can be provided to HCFA by using robust analysis and design techniques, not just coding, to build business applications. Although an AD tool can provide the means to build application software, the construction of models for analysis of the business problem and the design of a system solution are far more effective. To be effective as technology evolves and business requirements change, we must plan to build new business applications based upon analysis using carefully conceived design models that provide for the reuse of software components. Continuing to develop business applications without proper planning, analysis, and design will perpetuate a new generation of legacy systems that cannot easily adapt to changing business needs or exploit advances in technology. If this were to occur, key objectives and benefits of the ITA would not be fully realized.

I-CASE tools support the development of analysis and design models, providing for the use of those models in subsequent SDLC phases. I-CASE tools also help automate much of the SDLC, particularly with regard to the Architecture, Engineering, and Development categories of the tool selection framework (see Exhibit 4-19). Because of their superiority in keeping deliverables synchronized across the entire SDLC, I-CASE tools are preferred by HCFA in comparison to other CASE tools.

Upper-CASE tools, unlike I-CASE tools, lack the capabilities for automating the construction, testing, and implementation of business applications from their analysis and design models. Some upper-CASE tools use repositories that permit models to be imported into other development tools, thereby adding value to the analysis efforts. Upper-CASE tools can be used to complement VPEs and 3GLs. However, selecting an upper-CASE tool that is incompatible with a chosen VPE or 3GL can cause integration issues that offset many of the gains normally realized by the use of such complementary tools.

OOA/D tools parallel the functionality of traditional upper-CASE tools that support structured modeling techniques such as entity-relationship diagramming and data-flow diagramming. However, OOA/D products tend to employ radically different modeling techniques, including the

creation of integrated requirements/data/process models through use-case analysis, object modeling, state transitioning diagrams, and object interaction diagrams. Many of these tools maintain the traceability of model information from the requirements model through analysis and design. These tools also provide features to generate skeletal code in lower-level object-oriented languages such as C++, Ada, Smalltalk, Java, and some 4GLs. If OOA/D tools are being considered by HCFA for developing applications with object-oriented programming languages and 4GLs, then products that complement and integrate well with one another should be selected.

Object-oriented CASE tools available today generally do not provide the range of features and capabilities required for a comprehensive AD environment in a manner comparable with that of I-CASE tools. Features they tend to lack include complete code generation and software configuration management capabilities for enterprise deployment. Object-oriented CASE tools over time, however, may evolve into a complete functional replacement for traditional I-CASE tools.

HCFA's evaluation criteria should weight favorably tools that share a common repository through multiple phases of the SDLC, support roundtrip engineering, and allow iterative development using a common design model. We should select tools that support the target Application Architecture approach and preferably integrate well with tools used across SDLC phases, rather than select tools on the basis of best-of-breed functionality.

I-CASE tools that maintain application software at a high level of abstraction, such as models, are preferable to those that require maintenance at the physical-code level. Model-based systems engineering tools provide significant productivity and quality benefits to HCFA's overall AD efforts.

4.6.2.6 Development

HCFA's IT direction emphasizes the importance of rapid application development, maintainability of finished products, and the benefits of repeatable processes within a full life cycle development environment. Code construction tools range from high-end, design-driven code generators to low-end editing, debugging, and compiling products. HCFA must trade off the high productivity, low flexibility of the high-end tools with the low productivity, high flexibility of the low-end tools. Development tools that offer a hybrid approach to design integration for business rules logic, VPE for GUI development, and 3GL binding for specialized technical interface requirements to legacy systems are also preferred. OO tools, because of their natural modularity, ability to handle greater complexity, and potential for reuse, are preferred to structural and data-driven development approaches.

Full life cycle development tools must integrate with security and software configuration management facilities to speed development and allow construction by large teams of application developers. Realizing the problem of offering best-of-breed tools in all areas of the development life cycle, vendors are relying on integration with third-party tools that can specialize in staying current with these functions. Tools preferred by HCFA either must directly

interface to specific third-party products prescribed by our standards, or must not preclude the ability to do so using standard interface techniques.

Future application development within HCFA must be better, faster, and cheaper. Our target Application Architecture approach relies upon reusing generalized software processes and modules to speed up development while increasing quality and lowering costs over time. Formalized reuse is evolving from the strict common code sharing technique of the past to a feature-rich toolbox of reusable components for application construction under a modular, building block approach. Preferred development tools must not only support the reuse of their proprietary reusable assets, but also follow emerging standards that allow third-party vendors to offer reusable components. The tools must allow HCFA developers to create their own reusable module library that can also be made available to the enterprise as appropriate.

The capability to develop business applications for use on the Internet/Intranet is currently a driving force for tool selection, and is a mandatory criterion for HCFA. The ubiquitous-infrastructure and low-maintenance client support model offered by these Web-based technologies addresses the major issues in client/server application development that were encountered in the past.

HCFA will avoid using development tools that require client software other than standard browsers to be installed before the application will function, thereby reducing maintenance and support costs. Java and HTML forms are currently available choices. Distributed languages such as Java and Java Beans will continue to expand the capabilities of Web-based application development.

4.6.2.7 Operations

Tools in this category of the framework focus on criteria supporting the successful deployment of the developed application into a production environment. Operational capabilities are often overlooked as primary selection criteria when establishing an AD environment. Capabilities required by HCFA include system management services, software configuration management, and software distribution, from the installation of an application's first release to the point of its removal from production. With thorough investigation, it is possible to acquire fully functional AD tools without constraining the scalability or maintainability of the business applications developed using those tools.

Performance monitoring of an application assists in providing HCFA users with satisfactory response time. AD tools that support dynamic load balancing, which is driven or triggered by operation performance rules that are independent of application code, are highly preferred for use by HCFA.

Software configuration management and inventory control are mandatory operational capabilities that must be supported by AD tools chosen for use by HCFA. Security access control, authorization, and audit trail capabilities are also mandatory criteria. AD tools used by HCFA must either support these functions directly or provide standard interfaces to HCFA-specified third-party products providing these capabilities.

AD tools for HCFA must support software version control and rollout to the user environment. The automated control of application distribution is essential in a distributed computing environment. Criteria for HCFA to consider include the capability to capture inventory at the installation site; the capability to change execution environments; the capability to capture errors from failed installs and to rollback failed installs; and software metering.

4.6.2.8 Tool Selection

HCFA must select a comprehensive, integrated enterprise AD tool set from an array of commercially available products in a dynamic technology market. A more thorough evaluation of these tools will reveal their strengths and weaknesses in relation to supporting the target Application Architecture, and their impact on future AD efforts within HCFA. Identifying gaps or excessive overlaps in functionality will assist us in selecting the right combination of tools to meet HCFA's business needs.

To that end, HCFA has completed a study recommending a detailed approach to further evaluating enterprise modeling and AD tools for selection as an enterprise standard. The study sets the stage for detailed analysis and selection of tools by providing:

- An inventory of current HCFA enterprise tools;
- An evaluation of the current tools' suitability;
- Enterprise modeling tool selection criteria;
- Recommendations for further review and action by HCFA;
- Contact information for all tools reviewed; and
- A high-level framework describing the context for the use and support of enterprise modeling tools.

See Attachment C for the details of the study document. Further effort by HCFA is required in order to enable the selection of an appropriate enterprise AD tool set.

4.7 Policies

Policy is a central element of a sound architecture. Along with the IT Guiding Principles, IT Objectives, and technology standards, policies form the guideposts that will keep HCFA on the path to achieving the goals embodied in the IT Architecture. This section outlines policies specifically related to the Application Architecture. They are grouped based upon three levels of management focus: the linkage between applications and business requirements; project management and the use of standards; and system design. These policy statements were developed and agreed to during the HCFA Application Architecture Workshops held in December 1998.

4.7.1 Application/Business Requirements Policies

Applications should be designed and managed based upon the business functionality they are required to support. This begins with sound requirements-planning accomplished in the context of an enterprise-wide view of HCFA's business processes. Policy objectives in this area lead to a greater ability to link specific applications to their source requirements, and promote a more modularized approach to application design. The following policies apply:

Policy: Business requirements (rules) shall be clearly mapped to discrete business processes described in The Enterprise Business Function Model. Business rules shall include the identification of the trigger event for each business process.

Rationale: Business rules clearly mapped to discrete business processes allow changes to be easily identified and implemented. Such uniqueness prevents unplanned proliferation of software, increases software quality, minimizes the impact of change, and reduces cost.

Policy: The scope of an application should be targeted to the business process it supports. Business processes shall determine the logical boundaries for discrete application programs. Logical application boundaries shall remain inviolate.

Rationale: The scope of applications must be contained. Smaller, modular applications are easier to modify and adapt to change than are monolithic applications with multiple embedded business processes.

Policy: Application programs shall be logically allocated to an information system group. Application program boundaries shall not span information system group boundaries.

Rationale: Information system groups are the basis for evolving to an information-centric IT environment of well-managed databases. Organizing applications in this manner facilitates analysis and decision-making regarding tactical migration plans.

4.7.2 Project Management and Standards Policies

All application development projects across HCFA must adhere to common management policies and technology standards. Project management must ensure that tools selected for the development environment, and the standards used in the development and deployment of applications, have been properly adopted as part of the HCFA IT Architecture. Policies in this area provide a means to manage the diverse technology base that exists today, and foster planning for the development and deployment of business applications based upon target standards. The following policies apply:

Policy: Existing technologies will be grandfathered into the legacy standards category pending further consideration.

Rationale: Grandfathering a technology into the legacy standards category provides a starting point for further evaluation of the technology through a controlled process until target standards are identified. Standards categories are discussed in detail in Volume 5.

Policy: IT projects shall adhere to established management policies and standards for the acquisition of technology and tools used to support the design and development of business applications.

Rationale: Management policies and technology standards promote the desired use of technology by application developers, thereby supporting HCFA's continual evolution to the desired target Application Architecture state.

Policy: IT projects shall adhere to established standards and guidelines for the acquisition of technology to support deployment of business applications.

Rationale: Standards are necessary to enable an organized and consistent approach to IT management within HCFA. Adherence to standards optimizes the use of IT resources and reduces the complexity of the IT infrastructure, thereby maximizing the benefits to the broader HCFA enterprise.

4.7.3 Systems Design Policies

The policies in this section directly support the Design Principles identified in Section 4.4.1 of this document. The objective behind these policies is to reach the target Application Architecture by adhering to common design principles that result in modularized applications that are targeted to specific business processes, are more responsive to changing business needs, and are based upon HCFA's adopted technology standards. The following policies apply to this area:

Policy: IT projects shall adhere to the Application Architecture Design Principles for the design and development of business applications.

Rationale: The Application Architecture Design Principles provide specific guidance for how HCFA applications are to be designed and developed to achieve the desired target Application Architecture state.

Policy: Application programs shall be logically partitioned into layers consisting of separate user interface, business logic (rules), and database access program modules. Interfaces between application program modules from one layer to another shall be standardized. Application program module layers shall be location-independent to the extent possible.

Rationale: Modular programs can be reused by other business application processes. Location-independent modules can be deployed onto the infrastructure in an

optimal manner. This will facilitate changes to any one layer driven by changes in business needs.

Policy: Database access logic shall be designed as a separate layer of the program modules. The interface to the database access layer by the business logic layer shall be standardized. Database access layer modules shall be location-independent and accessible to any business application process that needs them.

Rationale: Separating database access logic minimizes the impact of changes. Standardized interfaces between layers simplify access to data by business application processes.

Policy: IT projects shall adhere to established technology policies and standards for the design and development of business applications.

Rationale: Policies and standards are necessary to enable an organized and consistent approach to application development within HCFA. Adherence to standards optimizes the use of IT resources, thereby maximizing the benefits to the broader HCFA enterprise.

4.8 Application Assessment and Migration

The HCFA application portfolio documents the logical information systems groups and the physical applications that are important to the enterprise, as well as the relationships of the physical applications to the Information Model and Business Function Model (BFM). The application portfolio analysis produced the following essential work products:

- Information Systems Groups to Business Function Matrix: Using the HCFA Information Systems Groups and the HCFA Business Function Model, an Information Systems Groups to Business Function Matrix was created to map applications to the business processes they support.
- Information Systems Groups to Physical Applications Matrix: Using the HCFA Information Systems Groups and the Physical Applications portfolio, an Information Systems Groups to Physical Applications Matrix was created to map physical applications to the information systems groups they support.
- Information Systems Groups to Subject Area Databases Matrix: Using the HCFA Information Systems Groups and the Subject Area Databases defined in the Information Architecture (Volume 3), an Information Systems Groups to Subject Area Databases Matrix was created to map information systems groups to the information databases they support.

The application portfolio contains several different views of the automation that is currently in place and that which will need to be in place to support HCFA's business. The collection of these views is intended to project a desired target application portfolio so that business

component and IT leaders will have a common vision for developing (or acquiring), integrating, and deploying application automation in a way that best supports HCFA's strategic business direction.

The HCFA Application Architecture and application portfolio will be used to:

- Define the HCFA Application Portfolio Quality Assessment – determine the functional and technical quality of the physical HCFA application portfolio.
- Validate the completeness of the HCFA Application Architecture.
- Identify gaps between HCFA legacy applications and HCFA target architectures.
- Define the Repository of Reusable Functional Modules – define the standard services and the interface standards for the “reuse” repository.
- Define migration paths for legacy applications that are not compliant with HCFA Application Architecture standards.

The process of documenting the application portfolio will require participation from functional components across HCFA. These activities will begin with development of the Migration Strategy, and continue as an ongoing part of maintaining and evolving the Application Architecture towards the target environment.

4.9 Feedback Form

A key indicator of the success of the ITA is feedback from the readers of this document. It is important that the ITA be responsive to the needs and objectives of those who are responsible for selecting technologies and deploying systems throughout HCFA. Your responses to this feedback form will allow HCFA staff to determine if this important mission of the ITA is being achieved. Please take a moment to complete the form and return it to the address indicated.

Do you feel this document is of value to you? Yes ____ No ____
Comments:

_____
_____
_____

Are there any topics of discussion that you feel should be added or changed? Yes___ No ___
Comments:

_____
_____

Please provide any other comments that you feel will improve the usability of this document:

_____
_____
_____

Please describe your job function:

_____
_____
_____

Please provide your name, mailing address, and E-mail address:

_____
_____
_____

Thank you for taking the time to provide your feedback on the HCFA ITA. Your comments and ideas are appreciated. Please send your completed form to the following address:

Sandy Haydock
HCFA IT Architecture Staff
7500 Security Boulevard, Mail Stop N3-15-25
Baltimore, MD 21244
shaydock@hcfa.go